Parul University

Faculty of Engineering & Technology
Subject Name : Fundamentals of Software Engineering
Subject Code : 203124254
B.Tech IT Year 2021-22 Semester 4th

FUNDAMENTALS OF SOFTWARE ENGINEERING

(203124254)

LAB MANUAL

INDEX

SR. NO.	TITLE	Number of hours	Page No.	Performance date	Assessment date	Marks out 0f 10	Sign
1	Project Definition and objective of the specified module and Perform RequirementEngineering Process.	2					
2	Identify Suitable Design and Implementation model from the different software engineering models.	2					
3	Prepare Software Requirement Specification (SRS) for the selected module.	2					
4	Develop Software project management planning (SPMP) for the specified module.	2					
5	Do Cost and Effort Estimation using different Software Cost Estimation models.	2					
6	Prepare System Analysis and System Design of identified Requirement specification using structure design as DFD with data dictionary and Structure chart for the specific module.	2					
7	Designing the module using Object Oriented approach including Use case Diagram with scenarios, Class Diagram and State Diagram, Collaboration Diagram, Sequence Diagram and Activity Diagram.	2					
8	Defining Coding Standards and walk through.	2					
9	Write the test cases for the identified module.	2					
10	Demonstrate the use of different Testing Tools with comparison.	2					
11	Define security and quality aspects of the identified module.	2					

PRACTICAL 1

AIM: Project Definition and objective of the specified module and Perform Requirement Engineering Process.

LIBER:

"LIBER" is the web-based application. It provides the users the information about the library in most handy way. It provides information about the books which are available at the library, provides online reading of books and also a facility to download the book online and many more features are also provide.

INNOVATIVE FEATURES OF THE SYSTEM:

- Mobile response web design which allows the user to access the same site from any type of device like smartphone, tablets, smart tv, pc, laptop.
- This web site contains new feature like "Help me find the book".
- Also provides the E-books at minimum charges.
- Also provides and interacting chat box where they are also given suggestion on the base of user's interest.
- Users can also read and write the reviews about the book.
- There is also a feature in which fine will be calculate automatically.
- Also tracks the user's record.

SCOPE OF THE SYSTEM:

- It only provides the information of the local library.
- Books include in the website will only be decided by administrator of the library.
- It focuses on:

- Online Reading of books
- Tracking whole record of User
- Calculate the fine of the users.

- It will be developed in PHP, CSS, HTML.
- This project manages the Library Resources.

MAIN MODULE OF THE SYSTEM:

- Admin Module.
- User Module.
- Search and view operation by site visitor (guest) for the books.
- Advertisement posting about the books.
- Payment of the books, fines etc.
- Facility of E-book.

REQUIREMENT OF SYSTEM:

DEFINITION OF REQUIREMENT ANALYSIS:

Analysis of system requirements involves a clear understanding of the application to be developed with the view of removing all ambiguities from user perception.

REQUIREMENT SPECIFICATION:

- **Non-functional requirement:**

  The non-functional requirements include those that are implicit and improve the quality of the software.

  - **Security:**

    There is a facility for security of data. Authenticated users can access the application.

  - **Easy to use:**

    The system is easy to use and a good GUI.

  - **Reliability:**

Reliability is assured by carrying out several tests for various test data as well as real live data and the output result matches the actual result.

The system has been tested thoroughly which is described in the testing part. Hence the system is reliable. The system supports generation of the printed reports.

- **Functional Requirement:**

1. USER LOGIN:

This feature used by the user to login into system. They are required to enter user id and password before they are allowed to enter the system.SThe user id and password will be verified and if invalid id is there user is allowed to not enter the system.

REQUIREMENTS:

- user id is provided when they register
- The system must only allow user with valid id and password to enter the system
- The system performs authorization process which decides what user level can access to.
- The user must be able to logout after they finished using system.

2. ADMINISTRATOR:

Administrator of the system has all the rights and authorities to view as well as tomodify and update the system whenever required for example he can add, remove, edit, area, valid zip code list, categories.

3. REGISTER A NEW USER:

This feature can be performed by all users to register new user to create account

REQUIREMENT:

- System must be able to verify information
- System must be able to delete information if information is wrong

4. REGISTER NEW BOOK:

This feature allows to add new books to the library.

REQUIREMENT:

- System must be able to verify information
- System must be able to enter number of copies into table.

- System must be able to not allow two books having same book id.

5. SEARCH BOOK:

This feature is found in book maintenance part. we can search book based on book id, book name, publication or by author name.

REQUIREMENT:

- System must be able to search the database based on select search type
- System must be able to filter book based on keyword entered
- System must be able to show the filtered book in table view

	VIEW	SEARCHING	MODIFY	ADD	WRITE REVIEWS
ADMINISTRATOR	Y	Y	Y	Y	N
GUEST	Y	Y	N	Y	Y

FEASIBILITY STUDY:

Feasibility study is the study of the application to check whether the application made is feasible or not. It is very useful to check whether the application works as per requirement or not. It is undertaken to determine the possibility of developing completely new applications. There are four feasibility studies that are considered.

- Technical feasibility
- Operational feasibility
- Implementation feasibility

- **TECHNICAL FEASIBILITY:**
  - It determines that work for the project is done with the present equipment and existing software technology.
  - As suggested for technical feasibility LIBER will be user friendly and has a better GUI.

- Does the proposed equipment have the technical capacity to hold the data required to use the new system?
- Are there technical guarantees of accuracy, reliability, ease of access and data security.
- **OPERATIONAL FEASIBILITY:**

  It covers mainly two aspects. It determines how the proposed system will fill in the current operation and what will happen if the job retraining and reconstructing may be needed at the end of the implementation system. The operational feasibility checks whether the user who is going to be using the system is able to work with the software in which the system is coded! System is very user friendly. Level of security and any other access control constraints are high.

- **IMPLEMENTATION FEASIBILITY:**
  - As we have mentioned that we are going to use PHP, HTML, CSS, MYSQL to developthis project, we found that these technologies are easy to learn and then use.
  - There is no copyright issue we face in development.
  - We can use any hardware configuration of pc/laptop to complete development of this project.
  - Once project is developed, it can be hosted on any web server with any operating system.

FEATURES OF THE SYSTEM:

- Also provides and interacting chat box where they are also given suggestion on the base of user's interest.
- Users can also read and write the reviews about the book.
- There is also a feature in which fine will be calculate automatically.
- Also tracks the user's record.

SYSTEM REQUIREMENT STUDY AND ASSUMPTION:

- **Hardware requirements**

The online system that we have built requires some specific hardware configuration; I have mentioned the basic minimum hardware recommendations to run the system adequately. Any higher configuration hardware would only add to the performance of the system. the minimum hardware would only add to the performance of the system. the minimum hardware requirements to run the system properly are as follows.

- 20 GB hard disk
- 512 DDR RAM
- Network card/network connection
- 1.4 GHZ processor

- **Software requirements**

➢ **Server side:**

- Operating System: any modern operating system like window, Linux, MACOS
- Web Server: Internet Information Services (IIS) Server or Apache.
- Database: MySQL
- Server-side scripting language: PHP

➢ **Client side:**

- Operating System: any modern operating system like window, Linux, MACOS
- Web Browser: any modern browser like Firefox Mozilla / Internet Explorer.

- **Technology used for development**
  - Front – End: HTML,CSS,JS,AJAX,BOOTSTRAP
  - Back – End: PHP,MYSQL
  - Tools: NotePad++/Sublime,Xamp
- **Operating System:**
  - Operating System: any modern operating system like window, Linux, MACOS.

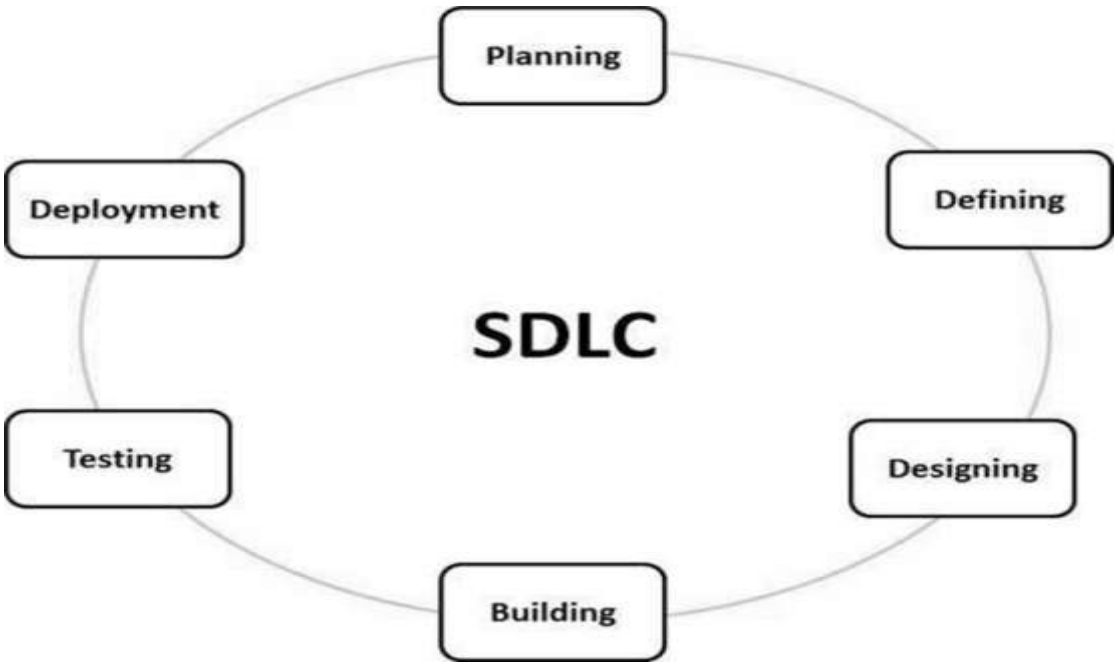
PRACTICAL 2

AIM: Identify Suitable Design and Implementation model from the different software engineering models.

SOFTWARE PROCESS MODELS:

- The process model is the abstract representation of process.
- Also known as Software development life cycle (SDLC) or Application development life cycle Models
- Process models prescribe a distinct set of activities, actions, tasks and milestones (deliverables) required to engineer high quality software.
- Process models are not perfect, but provide roadmap for software engineering work.
- Software models provide stability, control and organization to a process that if not managed can easily get out of control.
- Software process models are adapted (adjusted) to meet the needs of software engineers and managers for a specific project.

SDLC PHASES:



Stage 1: Planning and Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

Stage 2: Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an **SRS (Software Requirement Specification)** document which consists of all the product requirements to be designed and developed during the project life cycle.

Stage 3: Designing the Product Architecture

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

Stage 4: Building or Developing the Product

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

Stage 5: Testing the Product

This stage is usually a subset of all he stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

Stage 6: Deployment in the Market and Maintenance

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

LIST OF MODELS:

- Linear Sequential Waterfall model
- Prototype Model
- RAD Model.
- Evolutionary Software Process Model
    - Incremental Model
    - Spiral Model
    - Concurrent Development Process Model
    - Component based Assembly Model
- The Formal Methods Model
- Fourth Generation Techniques

DIFFERENCE BETWEEN MODELS:

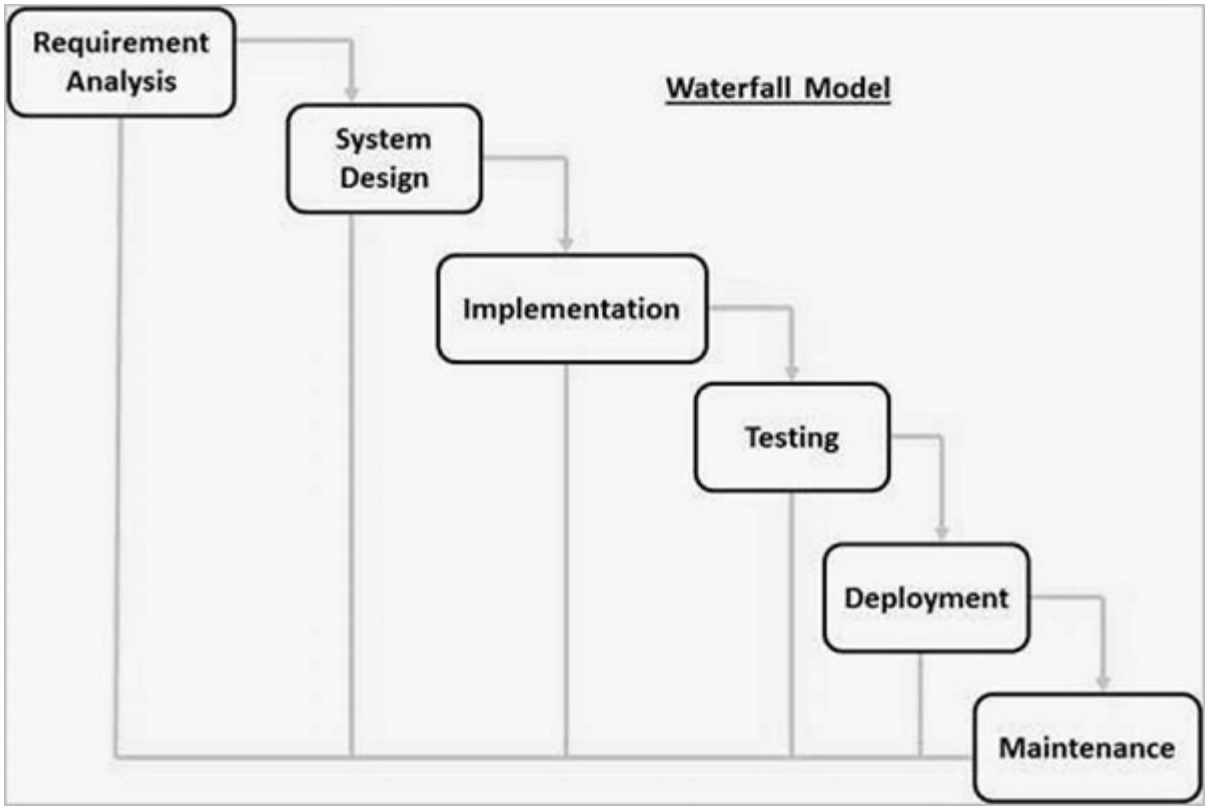
Properties of Model	Water-Fall Model	Incremental Model	Spiral Model	Rad Model
Planning in early stage	Yes	Yes	Yes	No
Returning to an earlier phase	No	Yes	Yes	Yes
Handle Large-Project	Not Appropriate	Not Appropriate	Appropriate	Not Appropriate
Detailed Documentation	Necessary	Yes but not much	Yes	Limited
Cost	Low	Low	Expensive	Low
Requirement Specifications	Beginning	Beginning	Beginning	Time boxed release
Flexibility to change	Difficult	Easy	Easy	Easy
User Involvement	Only at beginning	Intermediate	High	Only at the beginning
Maintenance	Least	Promotes Maintainability	Typical	Easily Maintained
Duration	Long	Very long	Long	Short
Risk Involvement	High	Low	Medium to high risk	Low
Framework Type	Linear	Linear + Iterative	Linear + Iterative	Linear
Testing	After completion of coding phase	After every iteration	At the end of the engineering phase	After completion of coding
Overlapping Phases	No	Yes (As parallel development is there)	No	Yes
Maintenance	Least Maintainable	Maintainable	Yes	Easily Maintainable
Re-usability	Least possible	To some extent	To some extent	Yes
Time-Frame	Very Long	Long	Long	Short
Working software availability	At the end of the life-cycle	At the end of every iteration	At the end of every iteration	At the end of the life cycle
Objective	High Assurance	Rapid Development	High Assurance	Rapid development
Team size	Large Team	Not Large Team	Large Team	Small Team
Customer control over administrator	Very Low	Yes	Yes	Yes

DIAGRAM FOR SELECTED MODEL AND ITDESCRIPTION:

- ## Waterfall Model

  - The waterfall model is the classical model of software engineering.
  - This model emphasizes planning in early stages, it ensures design flaws before they develop.
  - In addition, its intensive document and planning make it work well for projects in which quality control is a major concern.
  - When requirements for a problems are well understood then this model is used in which work flow from communication to deployment is linear
  - **When to use ?**
    - Requirements are very well known, clear and fixed
    - Product definition is stable
    - Technology is understood
    - There are no ambiguous (unclear) requirements
    - Ample (sufficient) resources with required expertise are available freely
    - The project is short.

DIAGRAM OF WATERFALL MODEL:



The sequential phases in Waterfall model are −

- **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

- **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

ADVANTAGES:

- Simple & Easy
- Easy to manage
- Works well for smaller projects
- Results are well documented

DRAWBACKS:

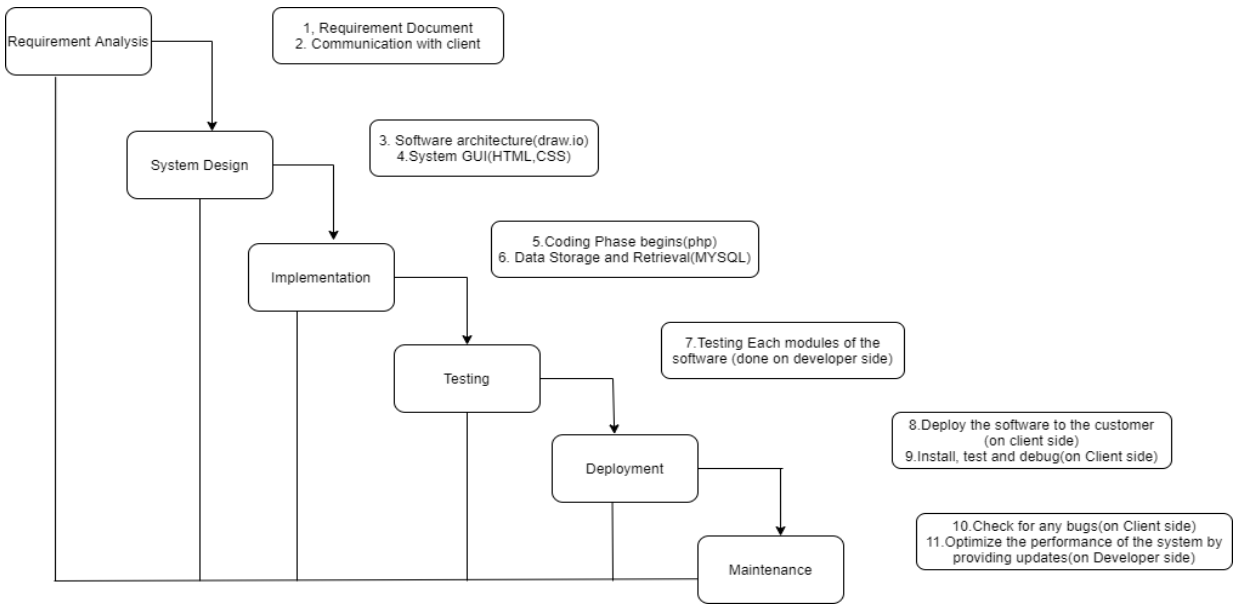
- Risk & Uncertainty
- Not for projects where requirements are changing
- Working version is not available during development. Which can lead the development with major mistakes.
- Deadlock can occur due to delay in any step.
- Not suitable for large projects.
- Not for big & complex projects

JUSTIFICATION: THAT WHY YOU CHOOSE THIS MODEL ONLY:

The Reason we choose waterfall model was:

- We are developing the project for small scale, and requirements are fixed i.e.no ambiguous (unclear) requirements.
- Results are well documented.
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- This model is simple to implement also the number of resources that are required for it is minimal.
- The requirements are simple and explicitly declared; they remain unchanged during the entire project development.
- The start and end points for each phase is fixed, which makes it easy to cover progress.
- The release date for the complete product, as well as its final cost, can be determined before development.
- It gives easy to control and clarity for the customer due to a strict reporting system.

DIAGRAM OF MODEL ACCORDING TO YOUR DEFINATION:



PRACTICAL 3

AIM:Study Software Requirement Engineering. Student should include SRS document for current semester project.

SRS(Softwarerequirementspecification):

- The production of the requirements stage of the software development process is Software Requirements Specifications (SRS) (also called a requirements document).
- This report lays a foundation for software engineering activities and is constructing when entire requirements are elicited and analyzed. SRS is a formal report, which acts as a representation of software that enables the customers to review whether it (SRS) is according to their requirements.
- Also, it comprises user requirements for a system as well as detailed specifications of the system requirements.
- The SRS is a specification for a specific software product, program, or set of applications that perform particular functions in a specific environment. It serves several goals depending on who is writing it.
- First, the SRS could be written by the client of a system. Second, the SRS could be written by a developer of the system.
- The two methods create entirely various situations and establish different purposes for the document altogether. The first case, SRS, is used to define the needs and expectation of the users. The second case, SRS, is written for various purposes and serves as a contract document between customer and developer

SRS should address:

The basic issues that the SRS shall address are the following:

a) Functionality. What is the software supposed to do?

b) External interfaces. How does the software interact with people, the system‘s hardware, other hardware, and other software?

c) Performance. What is the speed, availability, response time, recovery time of various • software functions, etc.?

d) Attributes. What are the portability, correctness, maintainability, security, etc. considerations?

e) Design constraints imposed on an implementation. Are there any required standards in effect, implementation language, policies for database integrity, resource limits, operating environment(s)

Name of System:

Library Management System

Assumptions:

Users are already registered

Requirement 1: Register

First the user will have to register/sign up. There are two different types 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 has to provide details about his/her name of address, phone number, email id.

Req1.1: 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.

Req 1.2: Login

- Input: Enter the membership number and password provided.
- Output: User will be able to use the features of software.

Requirement 2: Manage books by user.

Req 2.1: Book issued

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

Req 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.

Req 2.3: Issued 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

Req 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 error message will be send and if not then conformation message will be displayed.

Req 2.5: 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 the user can issue it.

Req 2.6: Return

- Input; Return the book to the library.
- Output: The issued list will be updated and the returned book will be listed out.

Req 2.7: 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 not renewed if then fine will be applied by Rs 10 per day.

Requirement 3: Manage book by librarian

Req 3.1 Update details of the books

Req 3.1.1 Add books

- Input: Enter the details of the books such as names, author, edition, quantity.
- Output: confirmation of addition.

Req 3.1.2 Remove books

- Input: Enter the name of the book and quantity of books.
- Output: Update the list of the books available.

Requirement 4: Payment

Req 4.1:Payment Gateway:

- Input: Card Details, Paytm Wallet.
- Output: User receives the message of debited amount and booking confirmation.

Requirement 5: Help me find a book

- Input: The customer will ask the admin, or users for the book
- Output: Admin, users will find the book and provide the link of the book.

Requirement 6: E-book

- Input: The user will ask for the e-book for the particular book.
- Output: The Admin will provide the E-book at minimal cost to the user.

- Processing: According to the book Prices set by the admin at the time of transaction the amount will be paid via the payment gateways.

Requirement 7: Security

- Passwords and value information is saved in database after hashing and AES encrypted, Admin has a separate login where he can add and modify user data if needed.

Requirement 8: Easy to use

- User Friendly.

Requirement 9: Reliability

- FAQ's

Requirement 10: Portability

- Website can be accessed on any device having any operating system.

PRACTICAL 4

AIM: Develop Software project management planning (SPMP) for the specified module

Project Scheduling & Tracking

It is an action that distributes estimated effort across the planned project duration, by allocating the effort to specific software engineering tasks

Scheduling Principles:

•Compartmentalization

•Interdependency

•Time Allocation

•Effort Validation

•Define Responsibilities

•Define Outcomes

•Define Milestones

Scheduling methods:

Two project scheduling methods that can be applied to software development.

- Program Evaluation and Review Technique (PERT)
- Critical Path Method (CPM)

Both techniques are driven by information already developed in earlier project planning activities:

- estimates of effort
- a decomposition of the product function
- the selection of the appropriate process model and task set
- decomposition of the tasks that are selected S

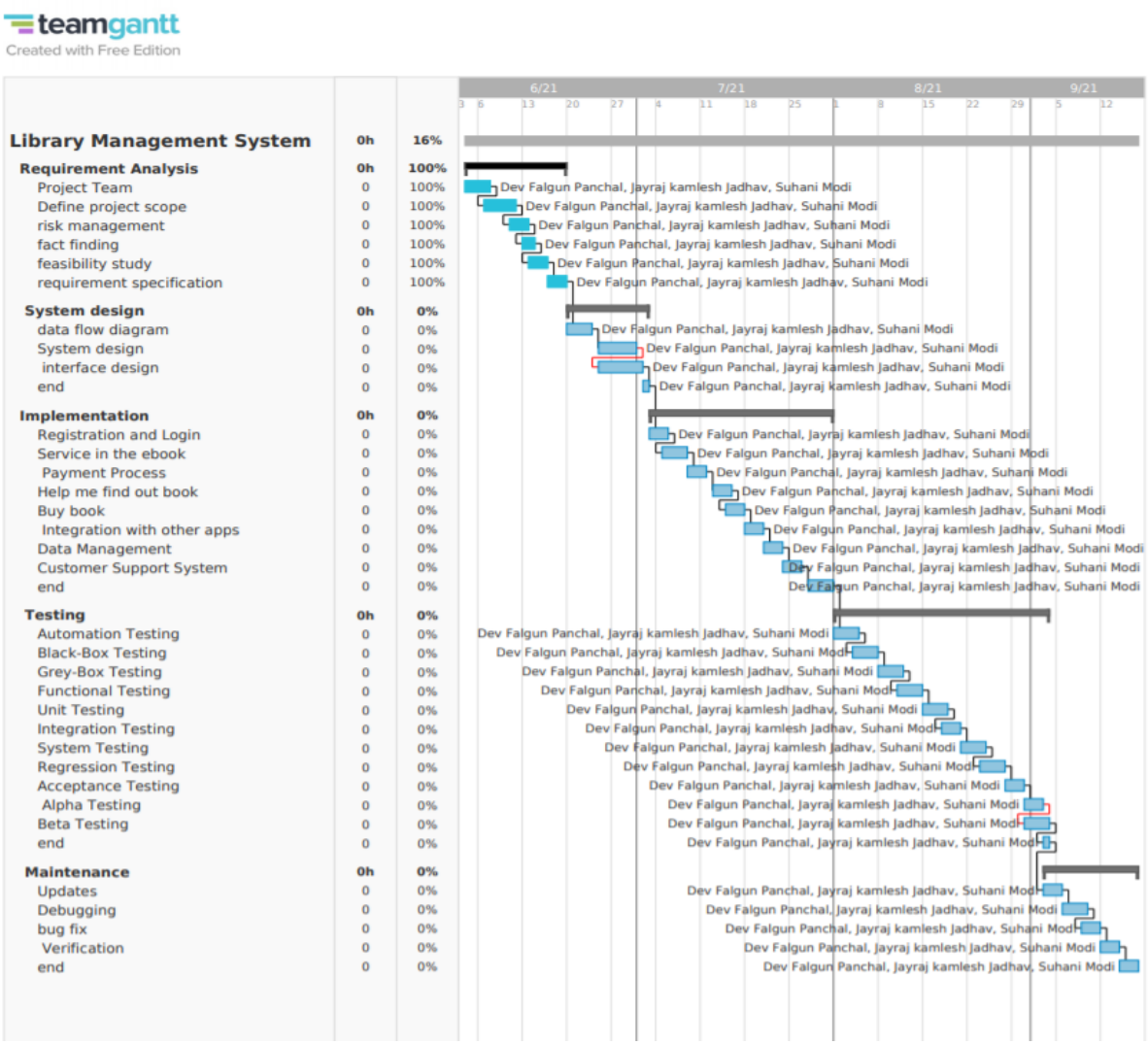
Gantt Chart:

A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time.

Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity. This allows you to see at a glance:

- What the various activities are
- When each activity begins and ends
- How long each activity is scheduled to last
- Where activities overlap with other activities, and by how much
- The start and end date of the whole project

GANTT CHART:



PRACTICAL 5

AIM:To make use of COCOMO model to find out the cost of software development.

Software Development Project:

Organic:

Application programs e.g. data processing programs.

A development project can be considered of organic type, if the project deals with developing a well understood application program, the size of the development team is reasonably small, and the team members are experienced in developing similar types of projects.

COCOMO Model:

COCOMO (Constructive Cost Estimation Model) was proposed by Boehm According to Boehm, software cost estimation should be done through three stages:

• Basic COCOMO,

• Intermediate COCOMO

• Complete COCOMO

Basic COCOMO Model:

The basic COCOMO model gives an approximate estimate of the project parameters The basic COCOMO estimation model is given by the following expressions

- KLOC is the estimated size of the software product expressed in Kilo Lines of Code,
- a1 , a2 , b1 , b2 are constants for each category of software products,
- Tdev is the estimated time to develop the software, expressed in months,
- Effort is the total effort required to develop the software product, expressed in person months (PMs).
- Every line of source text should be calculated as one LOC irrespective of the actual number of instructions on that line
- If a single instruction spans several lines (say n lines), it is considered to be nLOC

- The values of a1 , a2 , b1 , b2 for different categories of products (i.e. organic, semidetached, and embedded) as given by Boehm.

Calculation:

We have Assume that the size of an organic type software product has been estimated to be 3,000 lines of source code. Assume that the average salary of software engineers be Rs. 5,000/- per month. Determine the effort required to develop the software product and the nominal development time.

1. Effort = a1 * (KLOC)^ a2 PM
   = 2.4* (3) ^ 1.05 PM
   =7.6 PM
2. Tdev = b1*(Effort)^b2 Months
   = 2.5 * (7.6) ^ 0.38 Months
   =5.4 Months
3. Cost Required to develop the product = 5.4 * 5000

= Rs 27,000/-

4. Average staff = 7.6/5.4
   = 1.5 staff
5. Productivity = 3000 LOC/7.6 staff-months
   = 394.8 LOC/staff-month

2. Intermediate Model:

The basic Cocomo model assumes that the effort is only a function of the number of lines of code and some constants evaluated according to the different software system. However, in reality, no system's effort and schedule can be solely calculated on the basis of Lines of Code. For that, various other factors such as reliability, experience, Capability. These factors are known as Cost Drivers and the Intermediate Model utilizes 15 such drivers for cost estimation.

Cost drivers:

- Intermediate COCOMO introduces cost drivers.
- They are used because

  They are statistically significant to the cost of the project

  They are not correlated to the project size (KLOC).

Classifications of Cost drivers:

1. Product Attributes
2. Computer Attributes
3. Personnel Attributes
4. Project Attributes

- **Product Attributes:**
  - RELY: Required Software Reliability
  - DATA: Data Base Size
  - CPLX: Product Complexity
- **Computer Attributes:**
  - TIME: Execution Time Constraint
  - STOR: Main Storage Constraint
  - VIRT: Virtual Machine Volatility
  - TURN: Computer Turnaround Time
- **Personnel Attributes:**
  - ACAP: Analyst Capability
  - AEXP: Application Experience
  - PCAP: Programming Capability
  - VEXP: Virtual Machine Experience
  - LEXP: Programming Language Experience
- **Project Attributes:**
  - MODP: Modern Programming Practices
  - TOOL: Use of Software Tools

- SCED: Required Development Schedule

Cost Driver	Rating	Effort Multiplier
RELY	Nominal	1.00
DATA	Low	0.94
CPLX	Very High	1.30
TIME	Nominal	1.00
VIRT	High	1.06
STOR	High	1.11
TURN	Nominal	1.00
ACAP	High	0.86
AEXP*	Nominal	1.00
PCAP	High	0.86
VEXP	Low	1.10
LEXP	Nominal	1.00
MODP	High	0.91
TOOL	Low	1.10
SCED	Nominal	1.00
	Effort adjustment factor	**1.17**

Calculation:

1. Effort = a (KLOC)^b * C
   = 2.4 * (3) ^ 1.05 * 1.17
   = 8.90 PM

Where

- E is the effort
- a and b are constants (as before)
- KLOC is thousands of lines of code
- C is the effort adjustment factor

Cost Estimation for software:

External Input:

Req 1

Req 1.1

Req 1.2

Req 2

Req 2.2

Req 2.3

Req 2.4

Req 2.5

Req 2.6

Req 4

Req 4.1

Req 6

External Output:

Req 2.1

Req 2.7

Req 3.1

Req 3.1.1

Req 3.1.2

No. of Inquiries:

req 5

No. of External Files:

database

No. of External Interfaces:

Client machine

Printer

Payment

Calculation of Function Point (FP):

No. of External Inputs: 12

No. of External Outputs: 5

No. of External Inquiries: 1

No. of External Files: 1

No. of External Interfaces: 3

Count Total = 12*3+5*5+1*3+1*7+3*10

Count Total = 101

Information Domain Value	Count		Weighting factor Simple	Average	Complex		
External Inputs (EIs)		×	3	4	6	=	
External Outputs (EOs)		×	4	5	7	=	
External Inquiries (EQs)		×	3	4	6	=	
Internal Logical Files (ILFs)		×	7	10	15	=	
External Interface Files (EIFs)		×	5	7	10	=	
Count total	→						

Value Adjustment Factors (VAF):

F1: Does the system require reliable backup & recovery?	5
F2: Are data communications required?	5
F3: Are there distributed processing functions?	1
F4: Is performance critical?	3
F5: Will the system run in an existing, heavily utilized operational environment?	0
F6: Does the system require online data entry?	2
F7: Does the online data entry require the input transaction to be built over multiple screens or operation?	3
F8: Are the master files updated online?	4
F9: Are the inputs, outputs, files or inquiries complex?	1
F10: Is the internal processing complex?	1
F11: Is the code designed to be reusable?	2
F12: Are the conversion and installation included in design?	1
F13: Is the system design for multiple installations in different organizations?	2
F14: Is the application designed to facilitate change and ease of use by the user?	4

Function Point:

FP = Count Total * [0.65 + 0.01 * ∑(VAF)]

FP = 101*(0.65+0.01*(34))

FP = 34.99

1 FP = 32 LOC in VISUAL BASIC so,

34.99 = 1,119.8 LOC

so, approximately 1 KLOC final size.

Efforts:

= 2.4 * (1) ^ 1.05

= 2.4 PM

Tdev:

= 2.5 * (2.4) ^ 0.38

= 3.48 Months

PRACTICAL 6

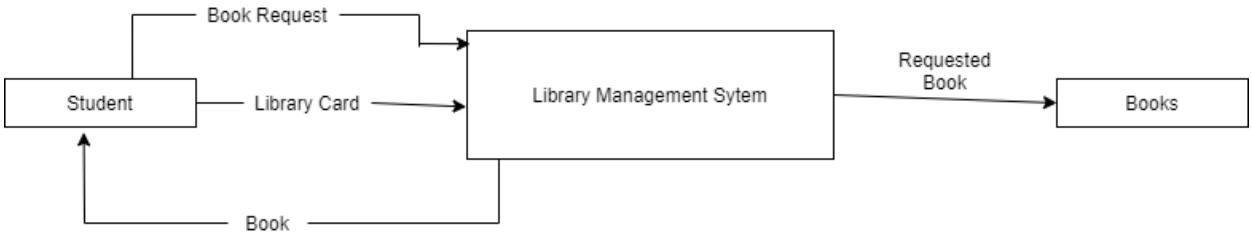
AIM: Prepare system analysis and system design of identified requirement specification using structure design as DFD with data dictionary for specific module.

DFD:

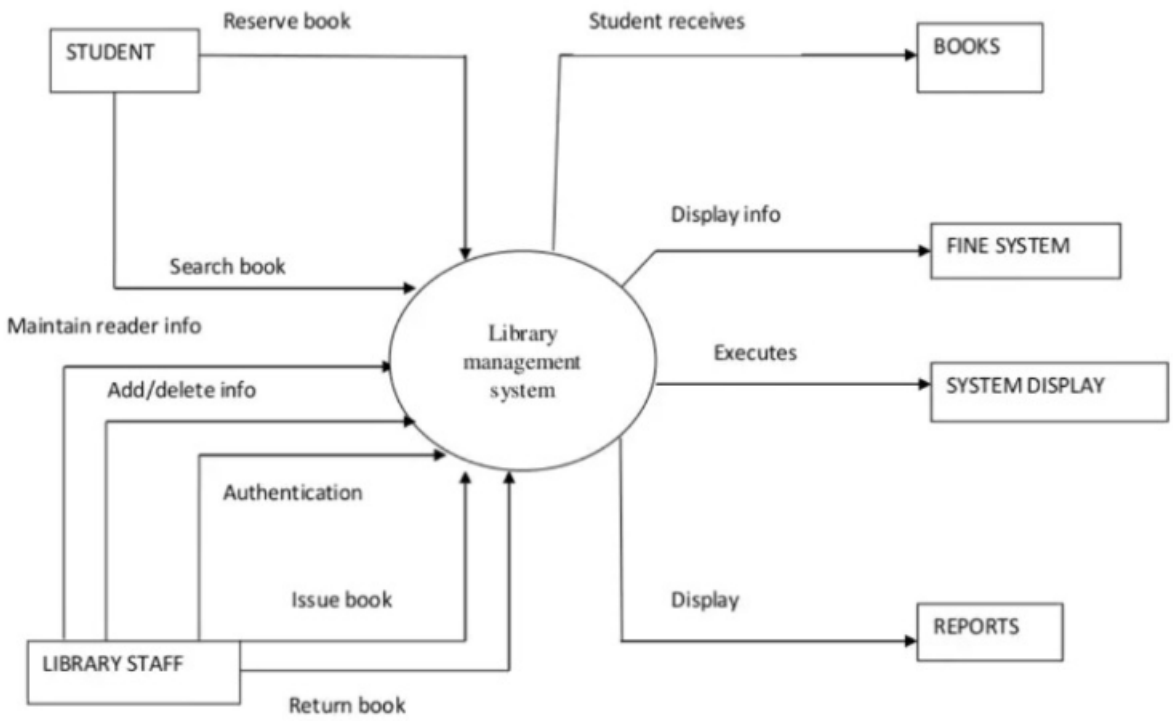
The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

Level 0 DFD(Context Level DFD):

At this level, the Input and Output of the system are shown. The system is designed and established across the world with input and output at this level.

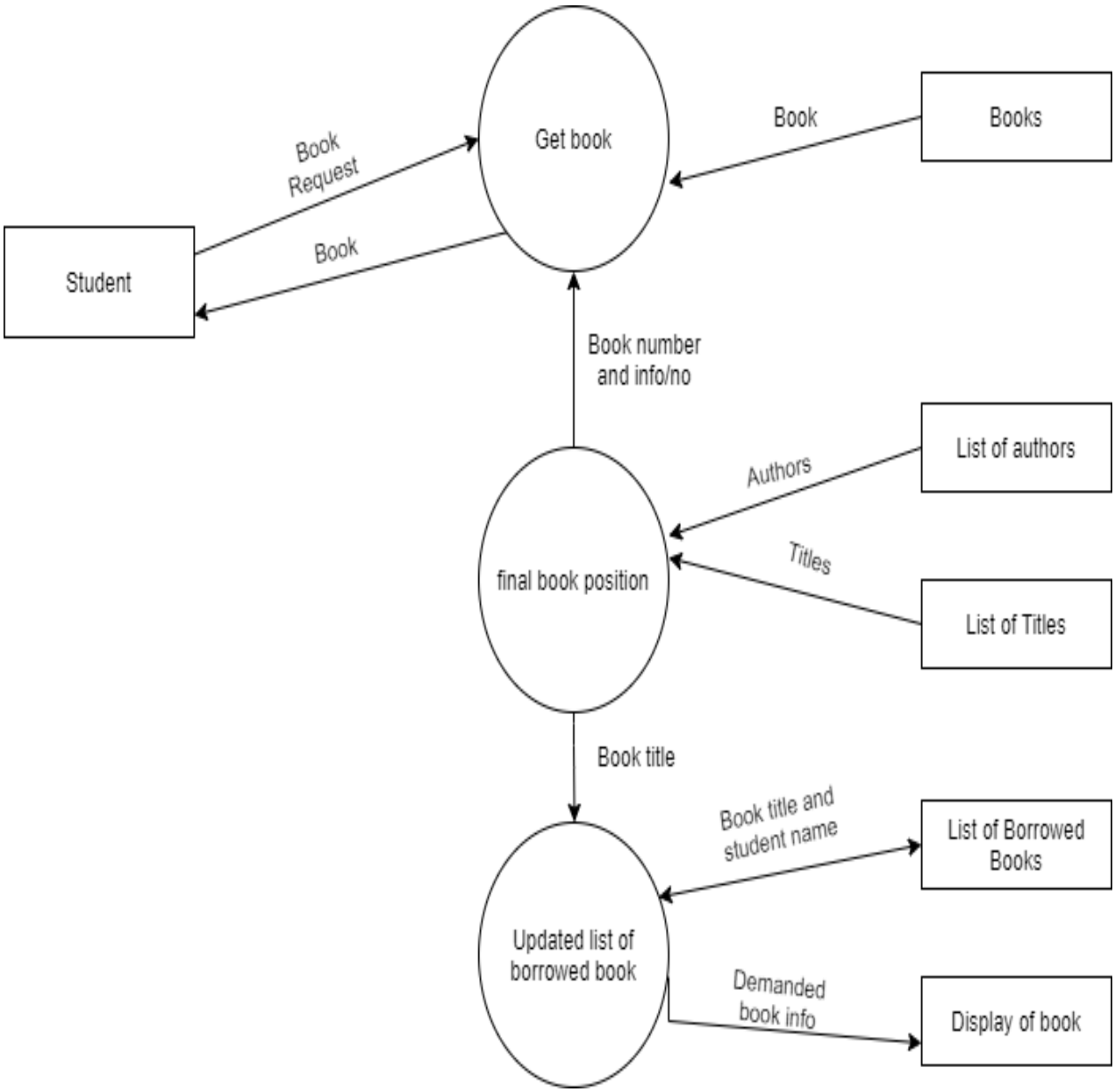


Level 1 DFD:


STUDENT
Reserve book
Student receives
BOOKS
Display info
FINE SYSTEM
Search book
Maintain reader info
Library management system
Executes
SYSTEM DISPLAY
Add/delete info
Authentication
Issue book
Display
REPORTS
LIBRARY STAFF
Return book

Level 2 DFD:

Detailed information about "Library management sytem" is shown below:



DATA DICTIONARY

- **LOGIN:**

Field name	Data type	Size	Constrain
Email	varchar	12	Not null
Password	Varchar	50	Not null
Id	Numeric	20	Primary key

- **REGISTRATION TABLE:**

Field name	Data type	Size	Constrains
Id	Numeric	-	Primary key
Email	Varchar2	255	Not null
Fullname	Varchar2	255	Not null
Mobail_Number	Varchar2	15	Not null
Password	Varchar2	255	Not null
Comfirm_Password	Varchar2	255	Not null

- **ADMIN:**

Field name	Data type	Size	Constrains
Admin_id	Varchar2	50	Primary key
Admin_pass	Varchar2	20	Not null
Admin_email	Varchar	10	Not null

- **MAKE PAYMENT:**

Field name	Data type	Size	Constrains
Payment_type	Varchar	10	Not null
Payment_amount	Numeric	15	Not null

- **ADD BOOKS:**

Field name	Data type	Size	Constrains
Book_name	Varchar2	30	Not null
Book_Id	Varcahr2	30	Not null
Author_name	Varchr2	10	-

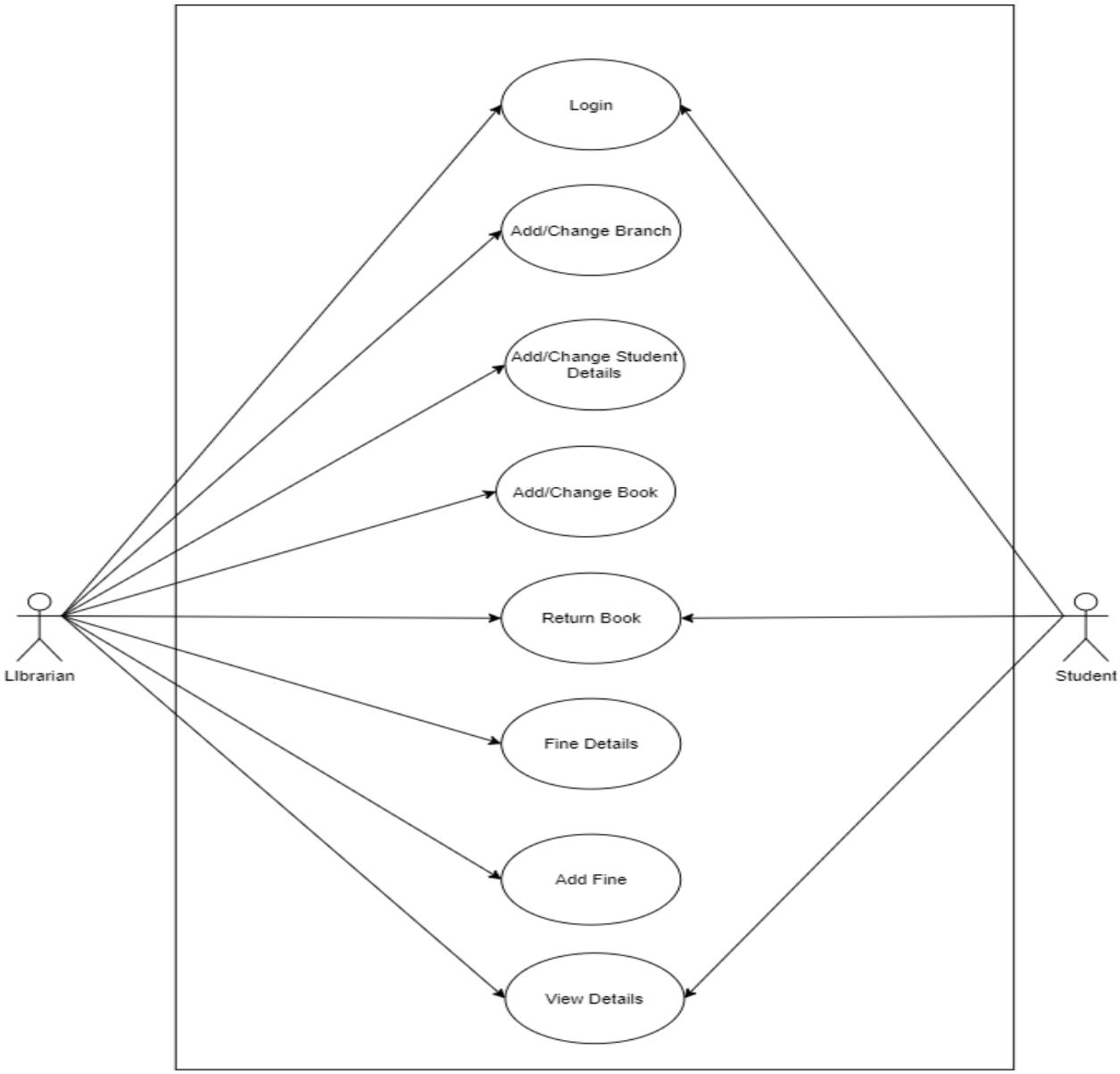
- **ISSUED BOOK:**

Field name	Data type	Size	Constrains
Student_name	Char	30	Not null
Student_Id	Numeric	10	Not null
Student_price	Numeric	-	Not null
Return	Varchar2	100	Not null
Dateofissue	Char	100	Not null

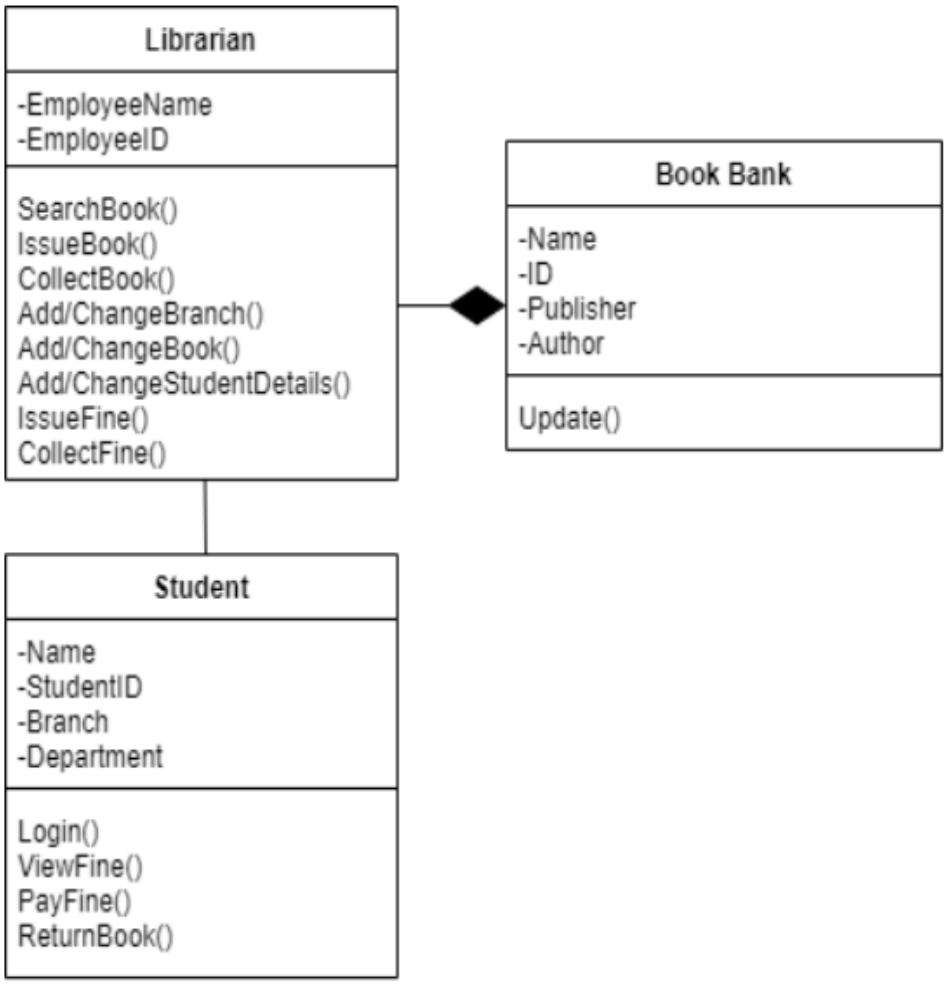
PRACTICAL 7

AIM:Designing the module using Object Oriented approach including Use case Diagram with scenarios, Class Diagram and State Diagram, Collaboration Diagram, Sequence Diagram and Activity Diagram.

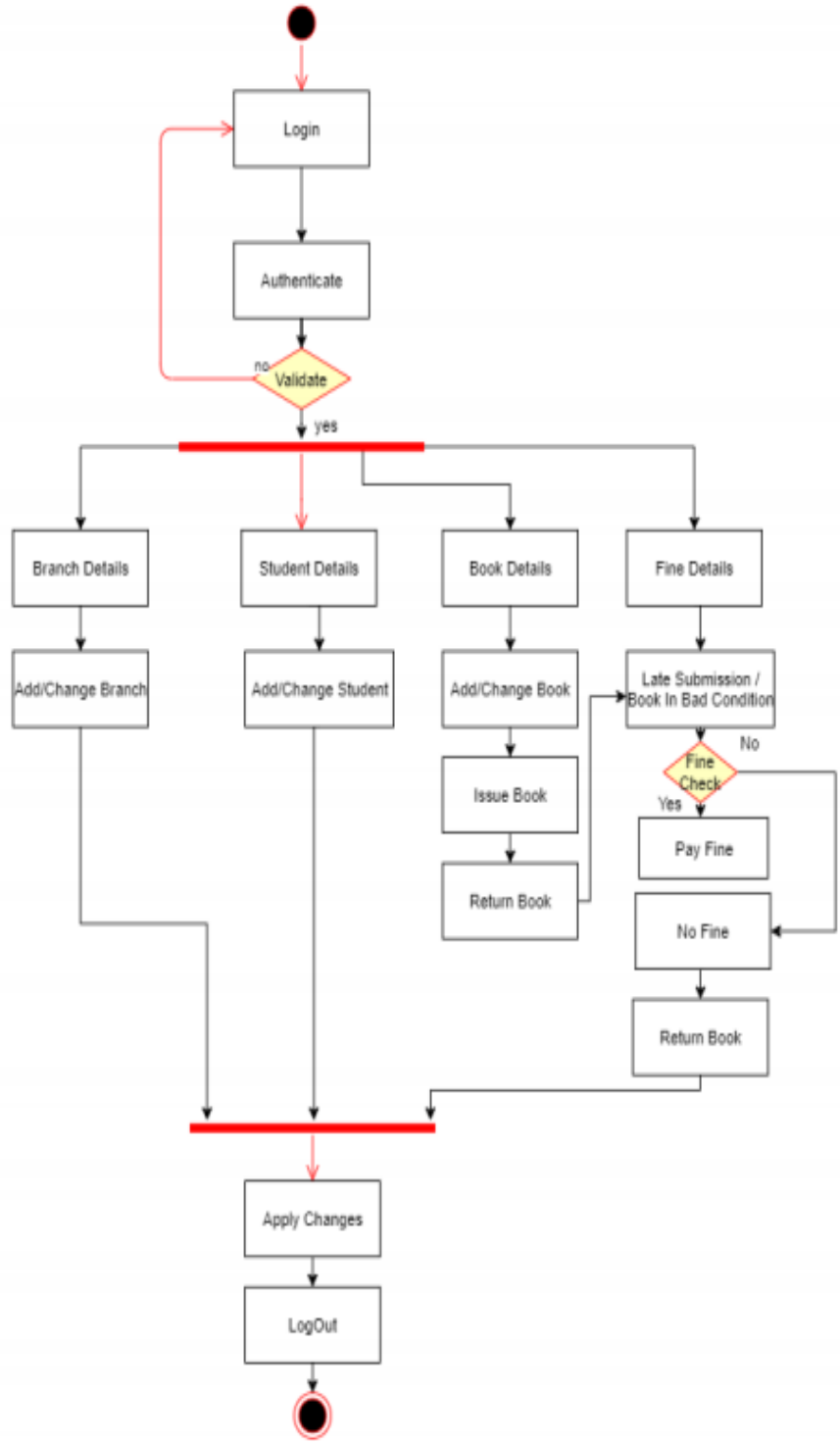
1. Use Case diagram:



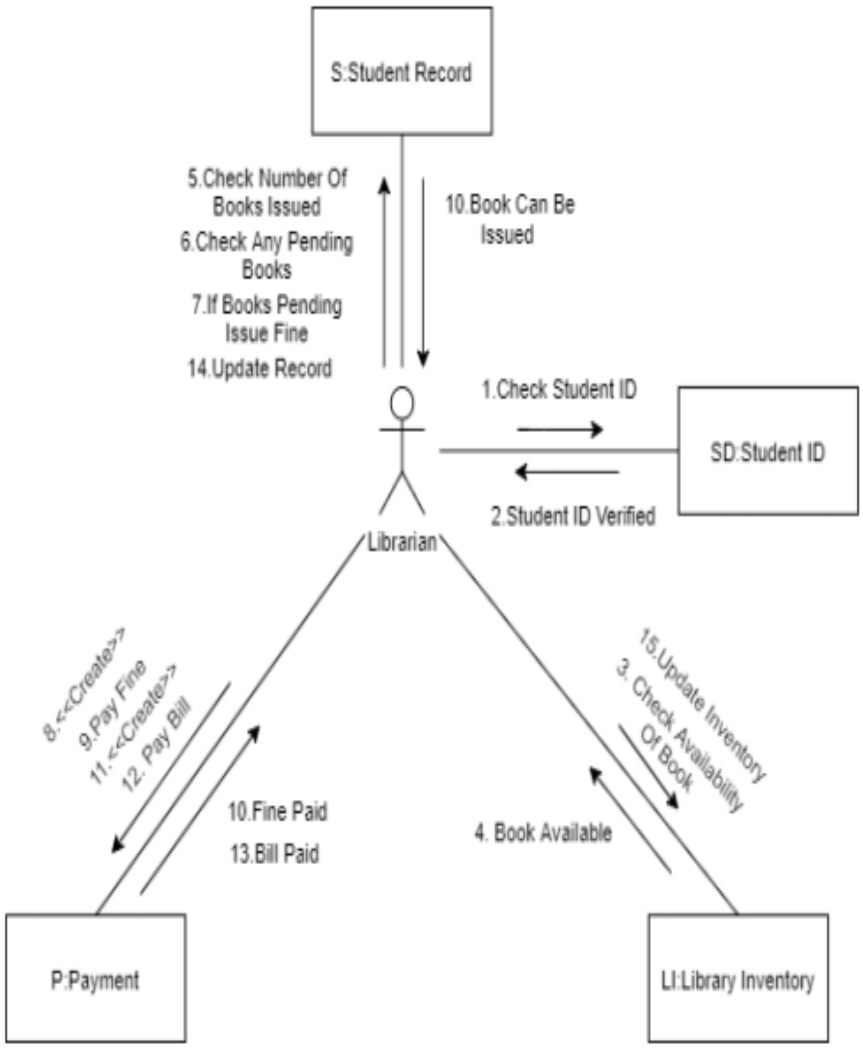
2. Class Diagram



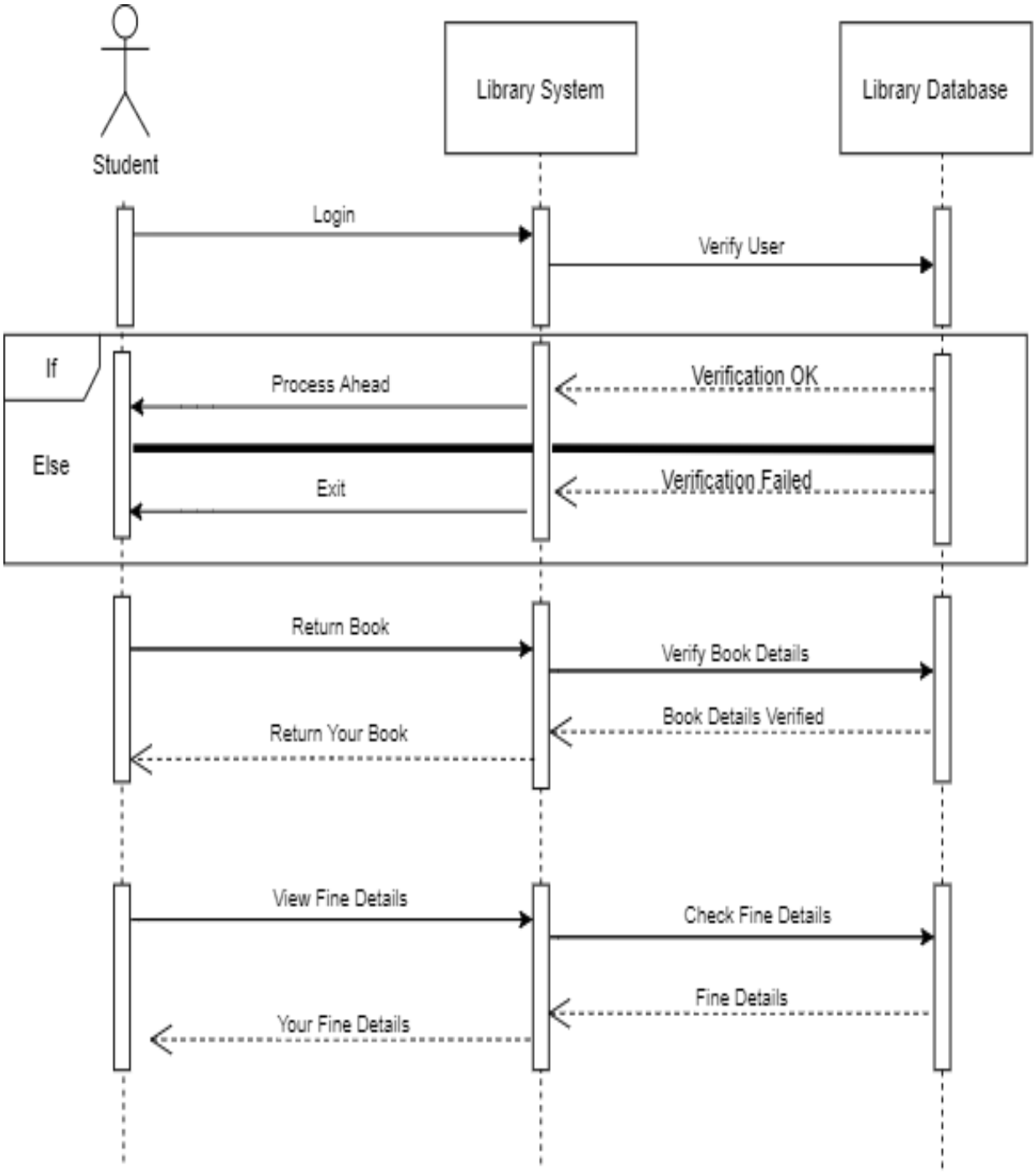
3. State Diagram



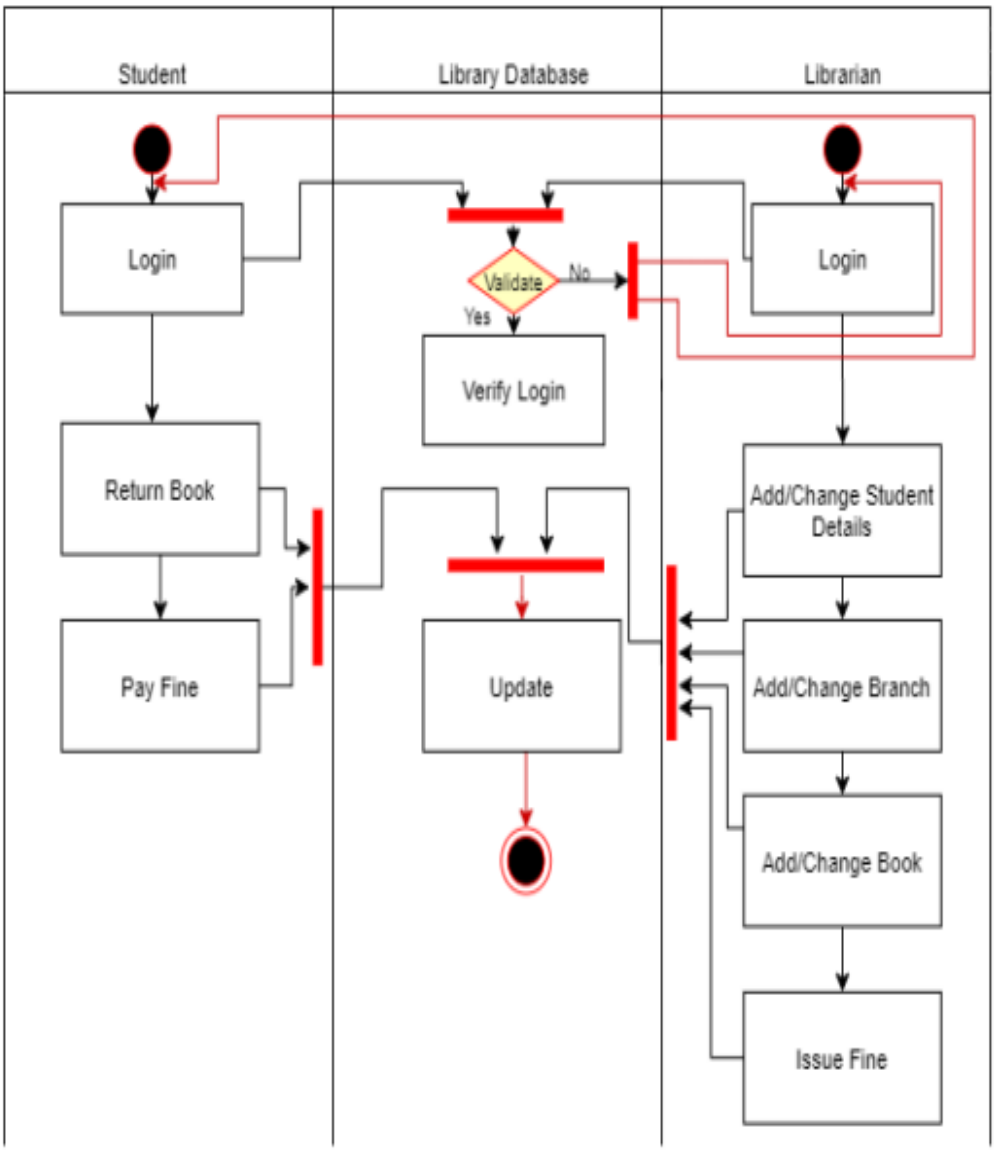
4. Collaboration Diagram



5.Sequence Diagram



6.Activity Diagram



PRACTICAL-8

AIM: Defining Coding Standards and walk through.

CodingStandards:

Different modules specified in the design document are coded in the Coding phase according to the module specification. The main goal of the coding phase is to code from the design document prepared after the design phase through a high-level language and then to unit test this code.

Good software development organizations want their programmers to maintain to some well-defined and standard style of coding called coding standards. They usually make their own coding standards and guidelines depending on what suits their organization best and based on the types of software they develop. It is very important for the programmers to maintain the coding standards otherwise the code will be rejected during code review.

Purpose of having coding standards:

- A coding standard gives a uniform appearance to the codes written by different engineers.
- It improves readability, and maintainability of the code and it reduces complexity also.
- It helps in code reuse and helps to detect error easily.
- It promotes sound programming practices and increases efficiency of the programmers.

Some of the coding standards:

1. **Limited use of globals:**

   These rules tell about which types of data that can be declared global and the data that can't be.

2. **Standard headers for different modules:**

   For better understanding and maintenance of the code, the header of different modules should follow some standard format and information. The header format must contain below things that is being used in various companies:

- Name of the module
- Date of module creation
- Author of the module
- Modification history
- Synopsis of the module about what the module does
- Different functions supported in the module along with their input output parameters
- Global variables accessed or modified by the module

3. Naming conventions for local variables, global variables, constants and functions:

Some of the naming conventions are given below:

- Meaningful and understandable variables name helps anyone to understand the reason of using it.
- Local variables should be named using camel case lettering starting with small letter (e.g. **localData**) whereas Global variables names should start with a capital letter (e.g. **GlobalData**). Constant names should be formed using capital letters only (e.g. **CONSDATA**).
- It is better to avoid the use of digits in variable names.
- The names of the function should be written in camel case starting with small letters.
- The name of the function must describe the reason of using the function clearly and briefly.

4.Indentation:

Proper indentation is very important to increase the readability of the code. For making the code readable, programmers should use White spaces properly. Some of the spacing conventions are given below:

- There must be a space after giving a comma between two function arguments.
- Each nested block should be properly indented and spaced.
- Proper Indentation should be there at the beginning and at the end of each block in the program.
- All braces should start from a new line and the code following the end of braces also start from a new line.

5. Error return values and exception handling conventions:

All functions that encountering an error condition should either return a 0 or 1 for simplifying the debugging.

On the other hand, Coding guidelines give some general suggestions regarding the coding style that to be followed for the betterment of understandability and readability of the code. Some of the coding guidelines are given below :

6.Avoid using a coding style that is too difficult to understand:

Code should be easily understandable. The complex code makes maintenance and debugging difficult and expensive.

7.Avoid using an identifier for multiple purposes:

Each variable should be given a descriptive and meaningful name indicating the reason behind using it. This is not possible if an identifier is used for multiple purposes and thus it can lead to confusion to the reader. Moreover, it leads to more difficulty during future enhancements.

8.Code should be well documented:

The code should be properly commented for understanding easily. Comments regarding the statements increase the understandability of the code.

9.Length of functions should not be very large:

Lengthy functions are very difficult to understand. That's why functions should be small enough to carry out small work and lengthy functions should be broken into small ones for completing small tasks.

10.Try not to use GOTO statement:

GOTO statement makes the program unstructured, thus it reduces the understandability of the program and also debugging becomes difficult.

Advantages of Coding Guidelines:

- Coding guidelines increase the efficiency of the software and reduces the development time.
- Coding guidelines help in detecting errors in the early phases, so it helps to reduce the extra cost incurred by the software project.
- If coding guidelines are maintained properly, then the software code increases readability and understandability thus it reduces the complexity of the code.
- It reduces the hidden cost for developing the software.

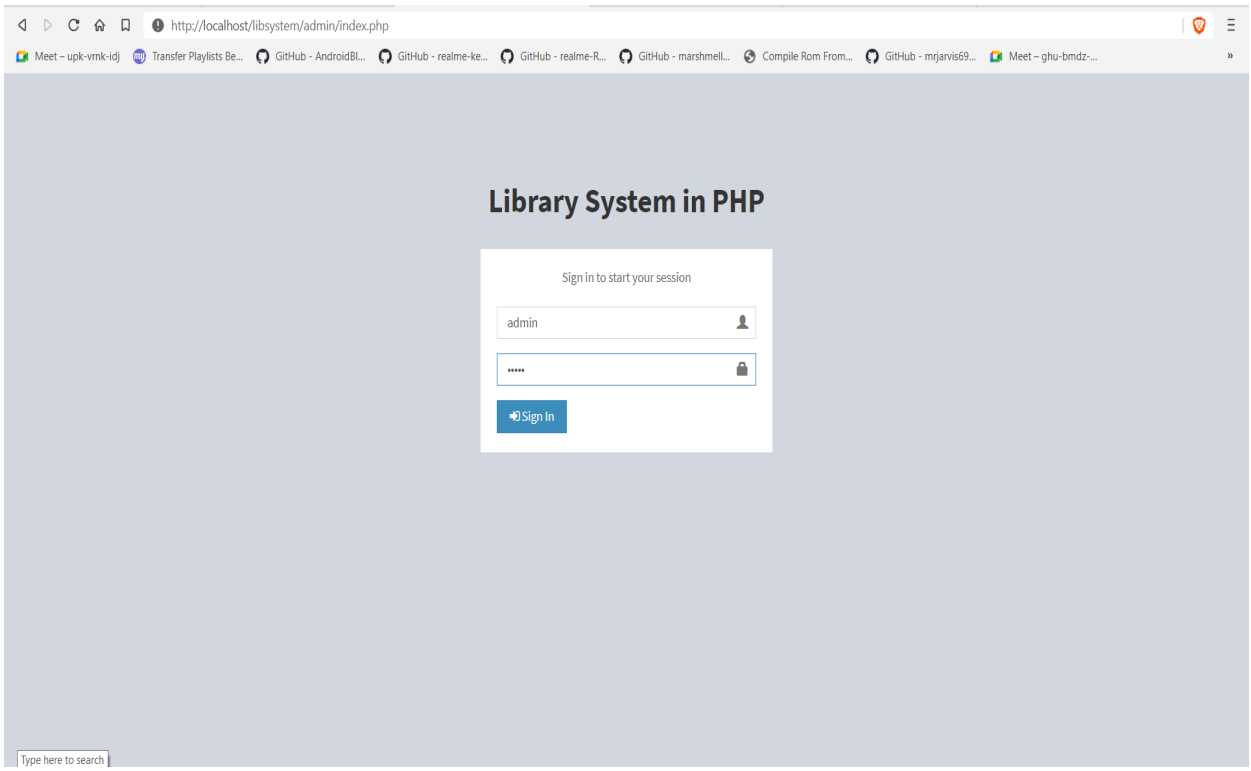
Practical 9

Aim: Write the test cases for the identified module.

LOGIN/LOGOUT:

ADMIN

OUTPUT:



LOGIN CODE:

```
<?php
session_start();
include 'includes/conn.php';
if(isset($_POST['login'])){
$username = $_POST['username'];
$password = $_POST['password'];
$sql = "SELECT * FROM admin WHERE username = '$username'";
$query = $conn->query($sql);
if($query->num_rows<1){
```

```
$_SESSION['error'] = 'Cannot find account with the username';
}
else{
$row = $query->fetch_assoc();
if(password_verify($password, $row['password'])){
$_SESSION['admin'] = $row['id'];
}
else{
$_SESSION['error'] = 'Incorrect password';
}
}}
else{
$_SESSION['error'] = 'Input admin credentials first';
}
header('location: index.php')
?>
```

LOGOUT CODE:

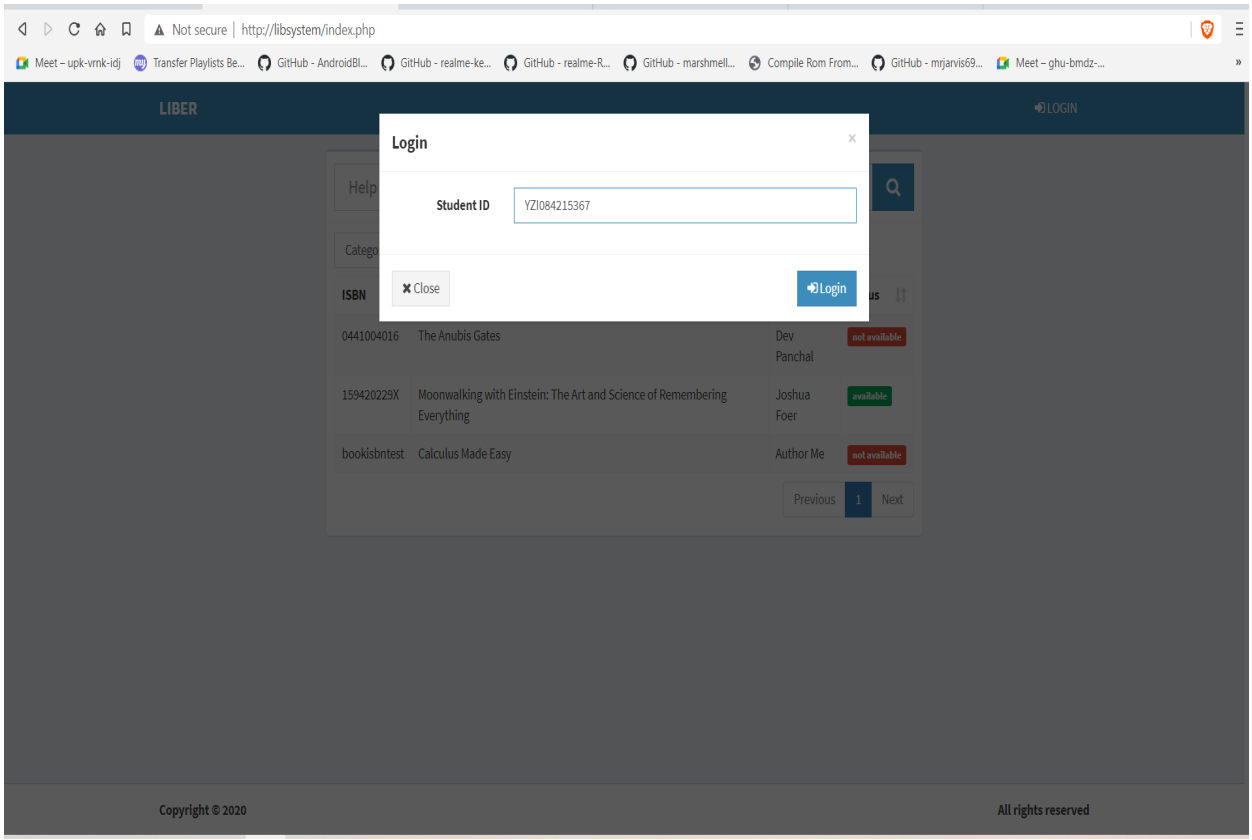
```
<?php
session_start();
session_destroy();
header('location: index.php');
?>
```

TESTCASE:

Test case ID	Test case	Test step	Test Data	Expected results	Actual results	pass/fail
01	Check Customer Login with valid Data	1. Go to site 2. Enter UserId 3. Enter Password 4. Click Submit	Username =admin Password =admin	congratulations	As expected	Pass
02	Check Customer Login with valid Data	1. Go to site 2. Enter UserId 3. Enter Password 4. Click Submit	Username =adminde mo Password =adminde mo	No such User exists. Invalid Credentials	As expected	Pass

STUDENT

OUTPUT:



LOGIN CODE:

```
<?php
include 'includes/session.php';
if(isset($_POST['login'])){
$student = $_POST['student'];
$sql = "SELECT * FROM students WHERE student_id = '$student'";
$query = $conn->query($sql);
if($query->num_rows>0){
$row = $query->fetch_assoc();
$_SESSION['student'] = $row['id'];
header('location: transaction.php');
}
else{
```

```
$_SESSION['error'] = 'Student not found';
header('location: index.php');
}}
else{
$_SESSION['error'] = 'Enter student id first';
header('location: index.php');
}
?>
```

LOGOUT CODE:

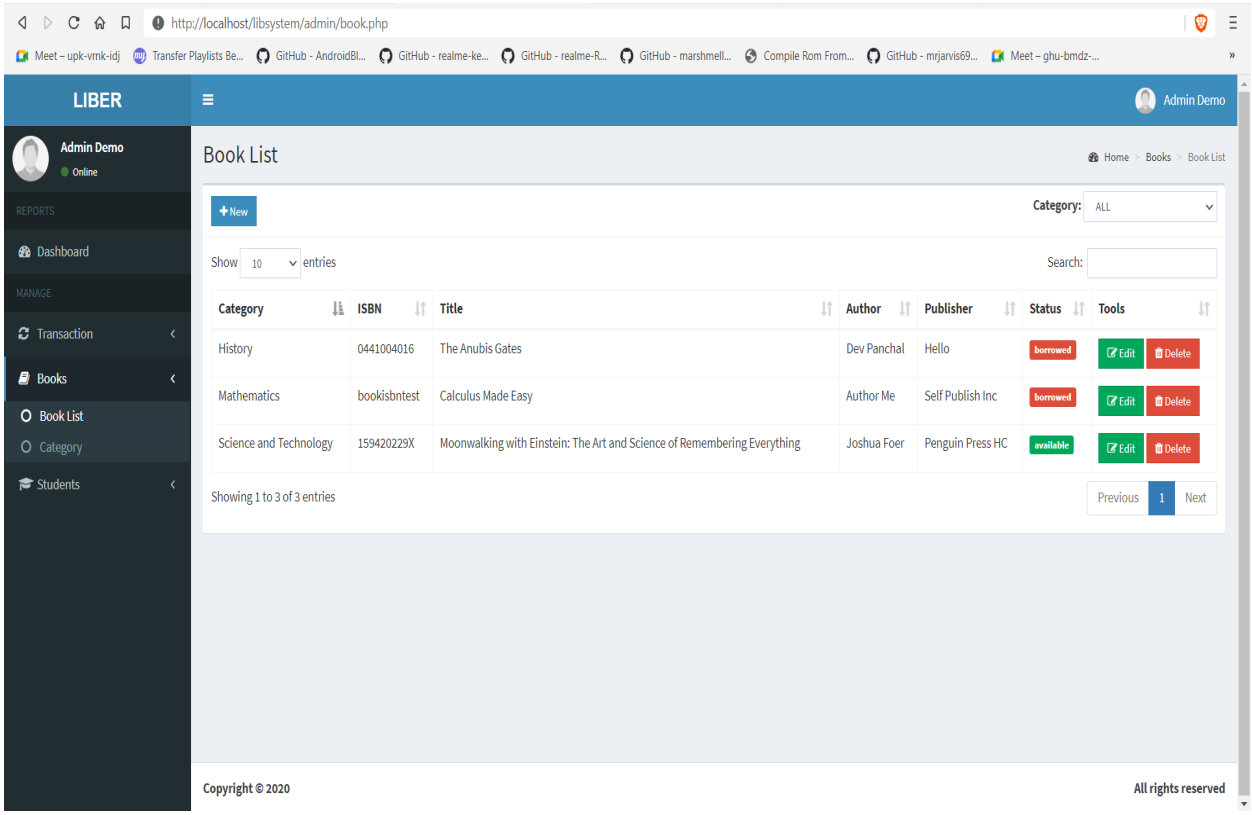
```
<?php
session_start();
session_destroy();
header('location: index.php');
?>
```

TESTCASE FOR LOGIN:

Test case ID	Test case	Test step	Test Data	Expected results	Actual results	pass/fail
01	Check Customer Login with valid Data	1. Go to site 2. Enter UserId 3.Enter Password 4.Click Submit	Student id = YZI084215367	congratulations	As expected	pass
02	Check Customer Login with valid Data	1. Go to site 2. Enter UserId 3.Enter Password 4.Click Submit	Student id = QRE123542346	No such User exists. Invalid Credentials	As expected	pass

BOOKS

OUTPUT:



BOOK LIST:

```
<?php include 'includes/session.php'; ?>
<?php
  $catid = 0;
  $where = '';
  if(isset($_GET['category'])){
    $catid = $_GET['category'];
    $where = 'WHERE books.category_id = '.$catid;
  }
?>
<?php include 'includes/header.php'; ?>
<body class="hold-transition skin-blue sidebar-mini">
<div class="wrapper">
```

```
<?php include 'includes/navbar.php'; ?>
<?php include 'includes/menubar.php'; ?>
<!-- Content Wrapper. Contains page content -->
<div class="content-wrapper">
<!-- Content Header (Page header) -->
<section class="content-header">
<h1>
      Book List
</h1>
<ol class="breadcrumb">
<li><a href="#"><i class="fa fa-dashboard"></i> Home</a></li>
<li>Books</li>
<li class="active">Book List</li>
</ol>
</section>
<!-- Main content -->
<section class="content">
<?php
      if(isset($_SESSION['error'])){
        echo "
<div class='alert alert-danger alert-dismissible'>
<button type='button' class='close' data-dismiss='alert' aria-hidden='true'>&times;</button>
<h4><i class='icon fa fa-warning'></i>Error!</h4>
".$_SESSION['error']."
</div>
        ";
        unset($_SESSION['error']);
      }
      if(isset($_SESSION['success'])){
        echo "
```

```
<div class='alert alert-success alert-dismissible'>
<button type='button' class='close' data-dismiss='alert' aria-hidden='true'>&times;</button>
<h4><i class='icon fa fa-check'></i>Success!</h4>
".$_SESSION['success']."
</div>
        ";
        unset($_SESSION['success']);
      }
    ?>
<div class="row">
<div class="col-xs-12">
<div class="box">
<div class="box-header with-border">
<a href="#addnew" data-toggle="modal" class="btnbtn-primary btn-smbtn-flat"><i class="fa fa-plus"></i> New</a>
<div class="box-tools pull-right">
<form class="form-inline">
<div class="form-group">
<label>Category: </label>
<select class="form-control input-sm" id="select_category">
<option value="0">ALL</option>
<?php
                    $sql = "SELECT * FROM category";
                    $query = $conn->query($sql);
while($catrow = $query->fetch_assoc()){
                     $selected = ($catid == $catrow['id']) ? " selected" : "";
                     echo "
<option value='".$catrow['id']."' ".$selected.">".$catrow['name']."</option>
                     ";
                    }
```

```
                ?>
</select>
</div>
</form>
</div>
</div>
<div class="box-body">
<table id="example1" class="table table-bordered">
<thead>
<th>Category</th>
<th>ISBN</th>
<th>Title</th>
<th>Author</th>
<th>Publisher</th>
<th>Status</th>
<th>Tools</th>
</thead>
<tbody>
<?php
                $sql = "SELECT *, books.id AS bookid FROM books LEFT JOIN category ON
category.id=books.category_id $where";
            $query = $conn->query($sql);
while($row = $query->fetch_assoc()){
                if($row['status']){
                  $status = '<span class="label label-danger">borrowed</span>';
                }
else{
                  $status = '<span class="label label-success">available</span>';
                }
                echo "
```

```
<tr>
<td>".$row['name']."</td>
<td>".$row['isbn']."</td>
<td>".$row['title']."</td>
<td>".$row['author']."</td>
<td>".$row['publisher']."</td>
<td>".$status."</td>
<td>
<button class='btnbtn-success btn-sm edit btn-flat' data-id='".$row['bookid']."'><i class='fa fa-edit'></i> Edit</button>
<button class='btnbtn-danger btn-sm delete btn-flat' data-id='".$row['bookid']."'><i class='fa fa-trash'></i> Delete</button>
</td>
</tr>
                ";
              }
            ?>
</tbody>
</table>
</div>
</div>
</div>
</div>
</section>
</div>
<?php include 'includes/footer.php'; ?>
<?php include 'includes/book_modal.php'; ?>
</div>
<?php include 'includes/scripts.php'; ?>
<script>
```

```
$(function(){
  $('#select_category').change(function(){
    var value = $(this).val();
if(value == 0){
window.location = 'book.php';
    }
else{
window.location = 'book.php?category='+value;
    }
  });
  $(document).on('click', '.edit', function(e){
e.preventDefault();
    $('#edit').modal('show');
    var id = $(this).data('id');
getRow(id);
  });
  $(document).on('click', '.edit', function(e){
e.preventDefault();
    $('#delete').modal('show');
    var id = $(this).data('id');
getRow(id);
  });
});
function getRow(id){
$.ajax({
    type: 'POST',
    url: 'book_row.php',
    data: {id:id},
dataType: 'json',
    success: function(response){
```

```
    $('.bookid').val(response.bookid);
    $('#edit_isbn').val(response.isbn);
    $('#edit_title').val(response.title);
    $('#catselect').val(response.category_id).html(response.name);
    $('#edit_author').val(response.author);
    $('#edit_publisher').val(response.publisher);
    $('#datepicker_edit').val(response.publish_date);
    $('#del_book').html(response.title);
   }
  });
}
</script>
</body>
</html>
```

BOOK_ADD:

```
<?php
include 'includes/session.php';
if(isset($_POST['add'])){
$isbn = $_POST['isbn'];
$title = $_POST['title'];
$category = $_POST['category'];
$author = $_POST['author'];
$publisher = $_POST['publisher'];
$pub_date = $_POST['pub_date'];
$sql = "INSERT INTO books (isbn, category_id, title, author, publisher, publish_date) VALUES
('$isbn', '$category', '$title', '$author', '$publisher', '$pub_date')";
if($conn->query($sql)){
$_SESSION['success'] = 'Book added successfully';
}
```

```
else{
$_SESSION['error'] = $conn->error;
}}
else{
$_SESSION['error'] = 'Fill up add form first';
}
header('location: book.php');
?>
```

BOOK_DELETE:

```
<?php
include 'includes/session.php';
if(isset($_POST['delete'])){
$id = $_POST['id'];
$sql = "DELETE FROM books WHERE id = '$id'";
if($conn->query($sql)){
$_SESSION['success'] = 'Book deleted successfully';
}
else{
$_SESSION['error'] = $conn->error;
}}
else{
$_SESSION['error'] = 'Select item to delete first';
}
header('location: book.php');
?>
```

BOOK_EDIT:

```
<?php
```

```
include 'includes/session.php';
if(isset($_POST['edit'])){
$id = $_POST['id'];
$isbn = $_POST['isbn'];
$title = $_POST['title'];
$category = $_POST['category'];
$author = $_POST['author'];
$publisher = $_POST['publisher'];
$pub_date = $_POST['pub_date'];
$sql = "UPDATE books SET isbn = '$isbn', title = '$title', category_id = '$category', author =
'$author', publisher = '$publisher', publish_date = '$pub_date' WHERE id = '$id'";
if($conn->query($sql)){
$_SESSION['success'] = 'Book updated successfully';
}
else{
$_SESSION['error'] = $conn->error;
}
}
else{
$_SESSION['error'] = 'Fill up edit form first';
}
header('location:book.php');
?>
```

BOOK ROW:

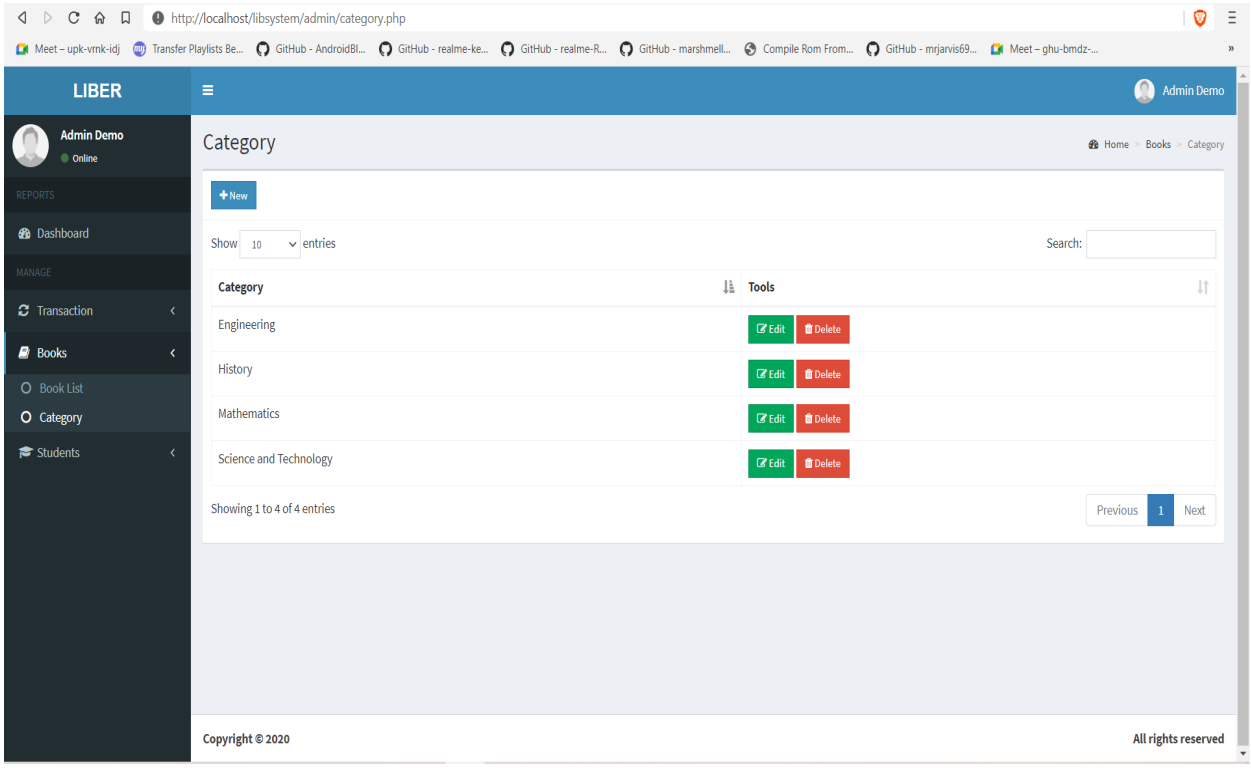
```
<?php
include 'includes/session.php';
if(isset($_POST['id'])){
$id = $_POST['id'];
$sql = "SELECT *, books.id AS bookid FROM books LEFT JOIN category ON category.id=books.category_id WHERE books.id = '$id'";
$query = $conn->query($sql);
$row = $query->fetch_assoc();
echo json_encode($row);
}
?>
```

TESTCASE FOR BOOK LIST:

Test case ID	Test case	Test step	Test Data	Expected results	Actual results	pass/fail
01	Check book list	1.Go to site 2. Enter title 3. Enter author 4.Enter ISBN 5.Enter Publisher 6.Click on submit	Category = Mathematics IBSN= Bookisbntest Title= Calculus Made Easy Author= Author Me Publisher= Self PublisherInc	congratulations	As expected	Pass
02	Check Book list	1.Go to site 2. Enter title 3. Enter author 4.Enter ISBN 5.Enter Publisher 6.Click on submit	Category = History IBSN= hellohiii Title= Friends Author= Sudha Murthy Publisher= Pigeon ltd.	No such book is found	As expected	Pass

CATEGORY

OUTPUT:



CODE:

```
<?php include 'includes/session.php'; ?>
<?php include 'includes/header.php'; ?>
<body class="hold-transition skin-blue sidebar-mini">
<div class="wrapper">

<?php include 'includes/navbar.php'; ?>
<?php include 'includes/menubar.php'; ?>

<!-- Content Wrapper. Contains page content -->
<div class="content-wrapper">
<!-- Content Header (Page header) -->
<section class="content-header">
<h1>
      Category
</h1>
```

```
<ol class="breadcrumb">
<li><a href="#"><i class="fa fa-dashboard"></i> Home</a></li>
<li>Books</li>
<li class="active">Category</li>
</ol>
</section>
<!-- Main content -->
<section class="content">
<?php
      if(isset($_SESSION['error'])){
        echo "
<div class='alert alert-danger alert-dismissible'>
<button type='button' class='close' data-dismiss='alert' aria-hidden='true'>&times;</button>
<h4><i class='icon fa fa-warning'></i>Error!</h4>
".$_SESSION['error']."
</div>
        ";
        unset($_SESSION['error']);
      }
      if(isset($_SESSION['success'])){
        echo "
<div class='alert alert-success alert-dismissible'>
<button type='button' class='close' data-dismiss='alert' aria-hidden='true'>&times;</button>
<h4><i class='icon fa fa-check'></i>Success!</h4>
".$_SESSION['success']."
</div>
        ";
        unset($_SESSION['success']);
      }
    ?>
```

```
<div class="row">
<div class="col-xs-12">
<div class="box">
<div class="box-header with-border">
<a href="#addnew" data-toggle="modal" class="btnbtn-primary btn-smbtn-flat"><i class="fa fa-plus"></i> New</a>
</div>
<div class="box-body">
<table id="example1" class="table table-bordered">
<thead>
<th>Category</th>
<th>Tools</th>
</thead>
<tbody>
<?php
                $sql = "SELECT * FROM category";
                $query = $conn->query($sql);
while($row = $query->fetch_assoc()){
                  echo "
<tr>
<td>".$row['name']."</td>
<td>
<button class='btnbtn-success btn-sm edit btn-flat' data-id='".$row['id']."'><i class='fa fa-edit'></i> Edit</button>
<button class='btnbtn-danger btn-sm delete btn-flat' data-id='".$row['id']."'><i class='fa fa-trash'></i> Delete</button>
</td>
</tr>
                  ";
                }
```

```
            ?>
</tbody>
</table>
</div>
</div>
</div>
</div>
</section>
</div>

<?php include 'includes/footer.php'; ?>
<?php include 'includes/category_modal.php'; ?>
</div>
<?php include 'includes/scripts.php'; ?>
<script>
$(function(){
  $(document).on('click', '.edit', function(e){
e.preventDefault();
    $('#edit').modal('show');
    var id = $(this).data('id');
getRow(id);
  });

  $(document).on('click', '.delete', function(e){
e.preventDefault();
    $('#delete').modal('show');
    var id = $(this).data('id');
getRow(id);
  });
});
```

```
function getRow(id){
$.ajax({
    type: 'POST',
    url: 'category_row.php',
    data: {id:id},
dataType: 'json',
    success: function(response){
      $('.catid').val(response.id);
      $('#edit_name').val(response.name);
      $('#del_cat').html(response.name);
    }
  });
}
</script>
</body>
</html>
```

CATEGORY ADD:

```
<?php
include 'includes/session.php';
if(isset($_POST['add'])){
$name = $_POST['name'];
$sql = "INSERT INTO category (name) VALUES ('$name')";
if($conn->query($sql)){
$_SESSION['success'] = 'Category added successfully';
}
else{
$_SESSION['error'] = $conn->error;
}
}
```

```
else{
$_SESSION['error'] = 'Fill up add form first';
}
header('location: category.php');
?>
```

CATEGORY DELETE:

```
<?php
include 'includes/session.php';
if(isset($_POST['delete'])){
$id = $_POST['id'];
$sql = "DELETE FROM category WHERE id = '$id'";
if($conn->query($sql)){
$_SESSION['success'] = 'Category deleted successfully';
}
else{
$_SESSION['error'] = $conn->error;
}
}
else{
$_SESSION['error'] = 'Select item to delete first';
}
header('location: category.php');
?>
```

CATEGORY EDIT:

```
<?php
include 'includes/session.php';
if(isset($_POST['edit'])){
```

```
$id = $_POST['id'];
$name = $_POST['name'];
$sql = "UPDATE category SET name = '$name' WHERE id = '$id'";
if($conn->query($sql)){
$_SESSION['success'] = 'Category updated successfully';
}
else{
$_SESSION['error'] = $conn->error;
}
}
else{
$_SESSION['error'] = 'Fill up edit form first';
}
header('location:category.php');
?>
```

CATEGORY ROW:

```
<?php
include 'includes/session.php';
if(isset($_POST['id'])){
$id = $_POST['id'];
$sql = "SELECT * FROM category WHERE id = '$id'";
$query = $conn->query($sql);
$row = $query->fetch_assoc();
echo json_encode($row);
}
?>
```

TESTCASE FOR CATEGORY:

Test case	Test case	Test step	Test Data	Expected results	Actual results	pass/fail

ID						
01	Check category	1. Go to site 2. Enter new category 3. Click on submit	Category = Mathematics	congratulations	As expected	Pass
02	Check Book list	1. Go to site 2. Enter new category 3. Click on submit	Category = Arts	No such Category is found	As expected	Pass

PRACTICAL 10

AIM:Define security and quality aspects of the identified module

QUALITY ASPECTS:

ISO 9001:

ISO 9001 sets out the criteria for a quality management system and is the only standard in the family that can be certified to (although this is not a requirement). It can be used by any organization, large or small, regardless of its field of activity. In fact, there are over one million companies and organizations in over 170 countries certified to ISO 9001. This standard is based on a number of quality management principles including a strong customer focus, the motivation and implication of top management, the process approach and continual improvement. Using ISO 9001:2015 helps ensure that customers get consistent, good quality products and services, which in turn brings many business benefits.

BENEFITS OF ISO:9001

An ISO 9001 Quality Management System (QMS) will help you streamline your processes, reduce errors, free up valuable management time and improve internal communications. Companies adopting this approach benefit from increased employee morale, improved customer retention and healthier revenues.

In short, through certification to ISO 9001 you demonstrate that your organisation is customer-focused and committed to delivering high quality services.

Here are the top benefits of using the ISO 9001 framework:

1. Increased efficiency: by following industry best-practice and focusing on quality you can reduce costs.
2. Increased revenue: through the reputation of ISO 9001 you can win more tenders and contracts, and by being more efficient you will also retain more customers and experience more repeat custom.
3. Higher levels of customer satisfaction: by understanding your customers' needs and reducing errors you increase customer confidence in your ability to deliver products and services.

4. Improved supplier relationships: because ISO 9001 certification ensures best-practice processes are in place which can contribute to more efficient supply chains, certification increases their confidence in your processes.
5. Improved employee morale: by improving internal communications you ensure everyone works to one agenda.

REQUIREMENTS OF ISO:9001:

a) needs to demonstrate its ability to consistently provide products and services that meet customer and applicable statutory and regulatory requirements, and

b) aims to enhance customer satisfaction through the effective application of the system, including processes for improvement of the system and the assurance of conformity to customer and applicable statutory and regulatory requirements.

All the requirements of ISO 9001:2015 are generic and are intended to be applicable to any organization, regardless of its type or size, or the products and services it provides.