PROJECT REPORT

BLOCKCHAIN TECHNOLOGY FOR POWERED LIBRARY MANAGEMENT SMART CONTRACT

Date	28 October 2023
Team ID	NM2023TMID01956
Project Name	BLOCKCHAIN TECHNOLOGY FOR BLOCKCHAIN-POWERED LIBRARY MANAGEMENT

TEAM MEMBERS

Anandharaj M	812720114312
Arun V	812720114319
Balachandar B	812720114321
Baskar B	812720114322

ABSTRACT 1. INTRODUCTION

Project Overview

ABSTRACT

Library management system is a project which aims in developing a computerized system to maintain all the daily work of library. This project has many features which are generally not available in normal library management system like facility of user login and a facility of teacher login. Now, it also has a facility of admin login through which the admin can monitor the whole system. It has also a facility where student after logging in their accounts can see list of books issued and its issues date and return date. It helps to maintain a database that is useful to enter new books and records of books borrowed by the members with the respective submission dates. It will reduce the manual work done by the librarian to maintain the record of the library. Using Barcode system this technology is helpful for librarian and reduce the valuable time. Using Barcode in libraries it saves library admin time by automatizing their task. This system involves installation of special software. A person using this system to borrow/return the books in library this system send a message notification to the particular user. Using this system, admin and users can avoid confusion & maintain the database correctly.

1. INTRODUCTION

"Blockchain-Powered Library Management" revolutionises traditional library systems by harnessing Ethereum smart contracts for transparent and secure book data management. This cutting edge approach ensures the integrity of library operations in a decentralised environment. Libraries,historical repositories of knowledge, can now seamlessly transition to a digital age with immutable and transparent book records stored on the blockchain. This system introduces a structured database where each book is represented by a smart contract, containing essential details such as title, author, ISBN, and ownership history. Users can query book information, and authorised personnel can efficiently add new books or transfer ownership with a single, secure transaction. By eliminating centralised intermediaries and enabling end-to-end verification, this system empowers libraries with unprecedented data transparency, security, and efficiency. Patrons can trust the accuracy of book details,while librarians can streamline operations and maintain an unforgeable history of book ownership changes."Blockchain-Powered Library Management" is the future of library administration,enhancing accessibility and trust in an ever-evolving digital landscape.

Project Overview

This is often a high-level description that captures the reader's interest. The aim is to change the ownership to another person.

Purpose

The purpose of the project is secured and high –level security to changing a ownership from one person to another person.

2. LITERATURE SURVEY

LIBRARY MANAGEMENT SYSTEMS – A SURVEY AUTHOR: Mrs. K. Sireesha

To keep track of library records, a library management system is used. It keeps track of the quantity of books in the library, how many books are issued, how many books are returned or renewed, and how much late fine money is owed, among other things. With this system, you can rapidly identify books, swiftly issue/reissue books, and handle all of the data in an efficient and organized manner. A library management system's goal is to deliver immediate and accurate information about any type of book, saving time and effort. The library management system is software that manages a library's manual functions. From preserving book records to issuing a book, the program aids in the management of the complete library business. It also facilitates the maintenance of fine details about books, such as the author's name, edition, and a variety of other crucial details. As a result, students and librarians will have an easier time searching for books and locating the appropriate materials. Electronic management using software is required to keep track of information such as issue date, due date, and who has borrowed any materials, among other things. The system was created and constructed with the goal of assisting schools and colleges in managing a modern library with correct data management. As a result, effective library management software is required to conduct smart school activities and keep correct library data. My Edu provides a smart school application that allows schools, colleges, etc,.

LIBRARY MANAGEMENT SYSTEM WITH TOPIC MODELLING AND ITS ADAPTABILITY TO OPEN AND DISTANCE LEARNING LIBRARIES

AUTHOR: Richard Adebayo

The use of libraries has grown tremendously in the last decade. Its processes, such as acquisition, cataloguing, shelving and the general management of information has evolved through the years, in aspects of digitisation and in fact knowledge management. All these would not have happened if not for the need for Librarians to make work easy for themselves and indeed the library users. This is one of the reasons Younis (2012) made aware that library users encounter problems when finding, borrowing, localising, renewing the borrowing, queuing for books. For most of the problems, solutions or ongoing researches are on to solve the problems. However, one area still short of in depth research,

is the area of integrating topic modelling into library management systems. Topic modeling is a kind of a probabilistic generative model that has been used widely in the field of computer science with a specific focus on text mining and information retrieval in recent years. The traditional means adopted by most users is to scan through each of the "suspected" books, before finally settling on one or two. This exercise can be tedious, tasking, time consuming and in fact, sometimes, ineffective. The digital age has exposed the minds of users into an unending world of possibilities, thereby, seeking new and more explorative ways of solving issues in library usage. One of these is topic modeling.

Existing problem

Begin by identifying and discussing the problems, challenges, or gaps in the current body of literature or in real-world scenarios that your project intends to address. This sets the stage for why your work is necessary. In the existing system all the transaction of books are done manually, So takes more time for a transaction like borrowing a book or returning a book and also for searching the books. Another major disadvantage is that to prepare the list of books borrowed and the available books in the library will take more time, currently it is done in one day for verifying all records. In this existing system takes lots of time for searching particular book in library.

References

1. N.J. Belkin and W.B. Croft, "Information Filtering and Information Retrieval: Two Sides of the Same Coin?" Comm.
2. P.W. Foltz and S.T. Dumais, "Personalized Information Delivery: An Analysis of Information Filtering Methods," Comm.
3. S. Pollock,"A Rule-Based Message Filtering System," ACM Trans. Office Information Systems.
4. Advanced .NET Remoting in VB.NET (Ingo Rammer, Apress, July 2002)
5. ASP to ASP.NET Migration Handbook (Christian Nagel et al, Wrox, January 2003)
6. Beginning Visual C# (Christian Nagel et al, Wrox, September 2001)
7. Data-Centric .NET Programming (Christian Nagel et al, Wrox, December 2001)

8. Professional .NET Network Programming 2nd Edition (Christian Nagel et al, Wrox, September 2004)

Problem Statement Definition

Clearly define the specific problem or research question that your project aims to solve. This problem statement should align with the gaps or issues you identified in the existing literature.

a. Inefficient Book Management: Tracking the availability, location, and condition of books is labor-intensive and prone to inaccuracies, leading to difficulties in locating specific books and managing inventory effectively.

b. Member Record Management: Managing member details, including borrowing history and fines, is time-consuming. Manual records often lead to errors and inconsistencies, affecting member services.

c. Ineffective Borrowing and Returning Process: The current system lacks automation in the borrowing and returning processes. This results in long queues, delays in issuing and returning books, and frustrated library patrons.

d. Limited Accessibility: Library resources are not accessible online, limiting the ability of users to search for books remotely. This lack of online catalog hampers user convenience and library outreach.

3. IDEATION & PROPOSED SOLUTION

Empathy Map Canvas

Ideation & Brainstorming
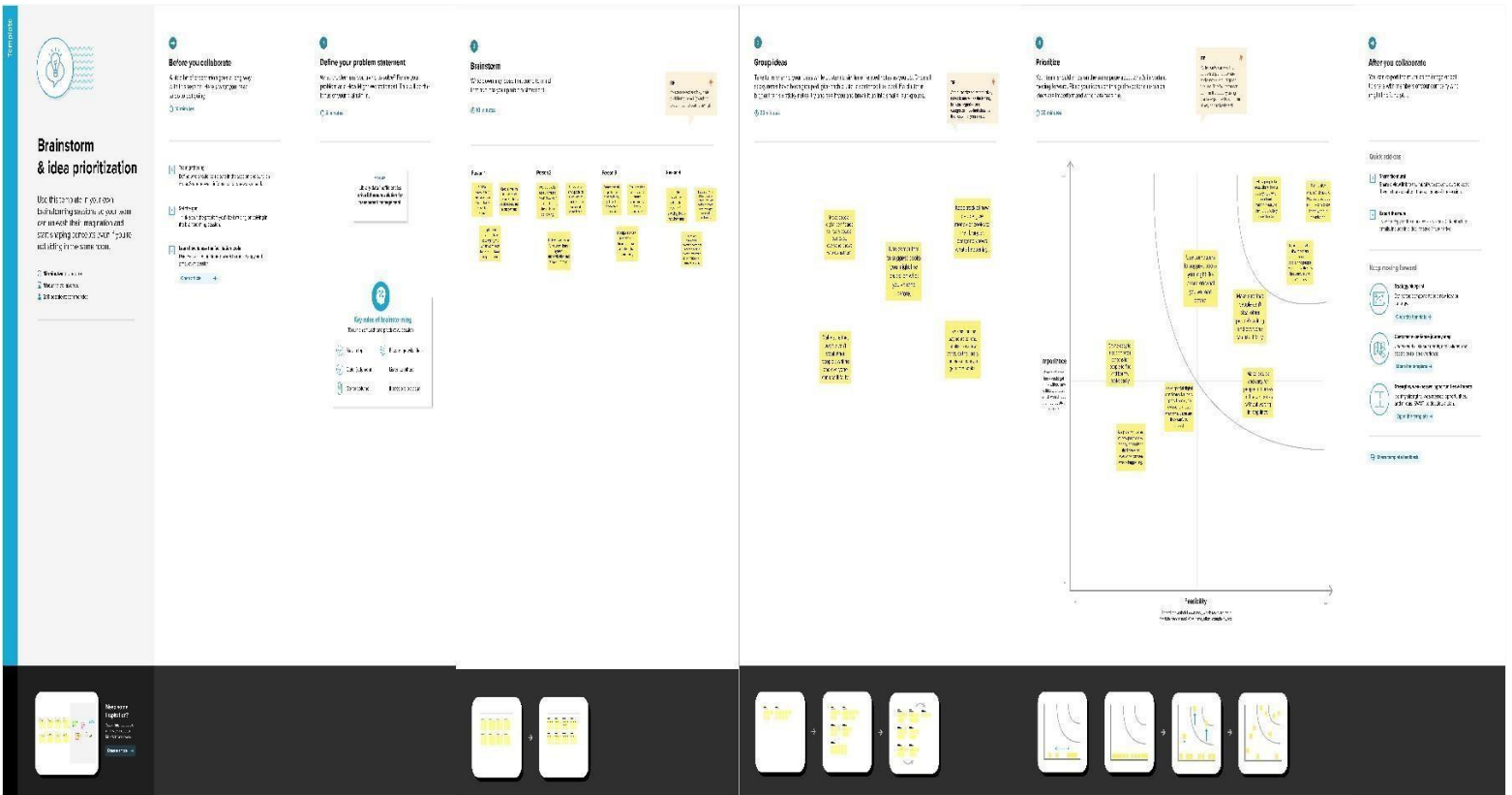
4.REQUIREMENT ANALYSIS

Functional requirement

Functional requirements define the specific behavior or functions of a system. They describe what the system should do and are often presented as specific features, capabilities, or interactions. Functional requirements are essentially the "what" of the system, outlining the services the system must provide.

Non-Functional requirements

Non-functional requirements, on the other hand, define the quality attributes, system performance, security, and overall user experience. They describe "how" the system performs certain functions rather than the functions themselves. Non-functional requirements are just as critical as functional requirements, as they directly impact user satisfaction and system usability.

5. PROJECT DESIGN

DATA FLOW DIAGRAM

A two-dimensional diagram explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a data flow diagram must identify external inputs and outputs, determine how the inputs and outputs relate to each other, and explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

LEVEL 0

The Level 0 DFD shows how the system is divided into 'sub-systems' (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.
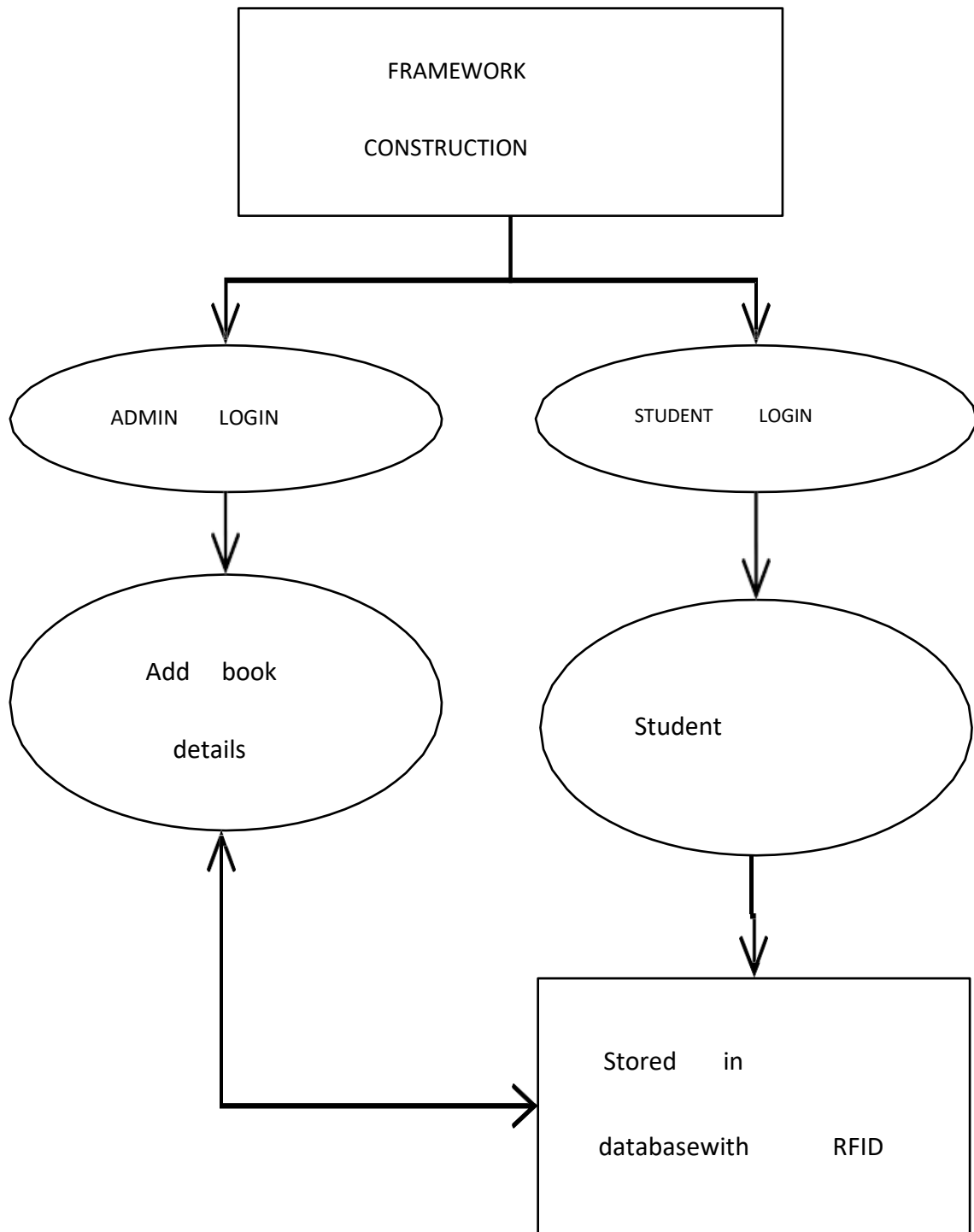
Figure No.5.1.1: Data Flow Diagram (level 0)

LEVEL 1

The next stage is to create the Level 1 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, to describe the system was using between two and seven functions two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.
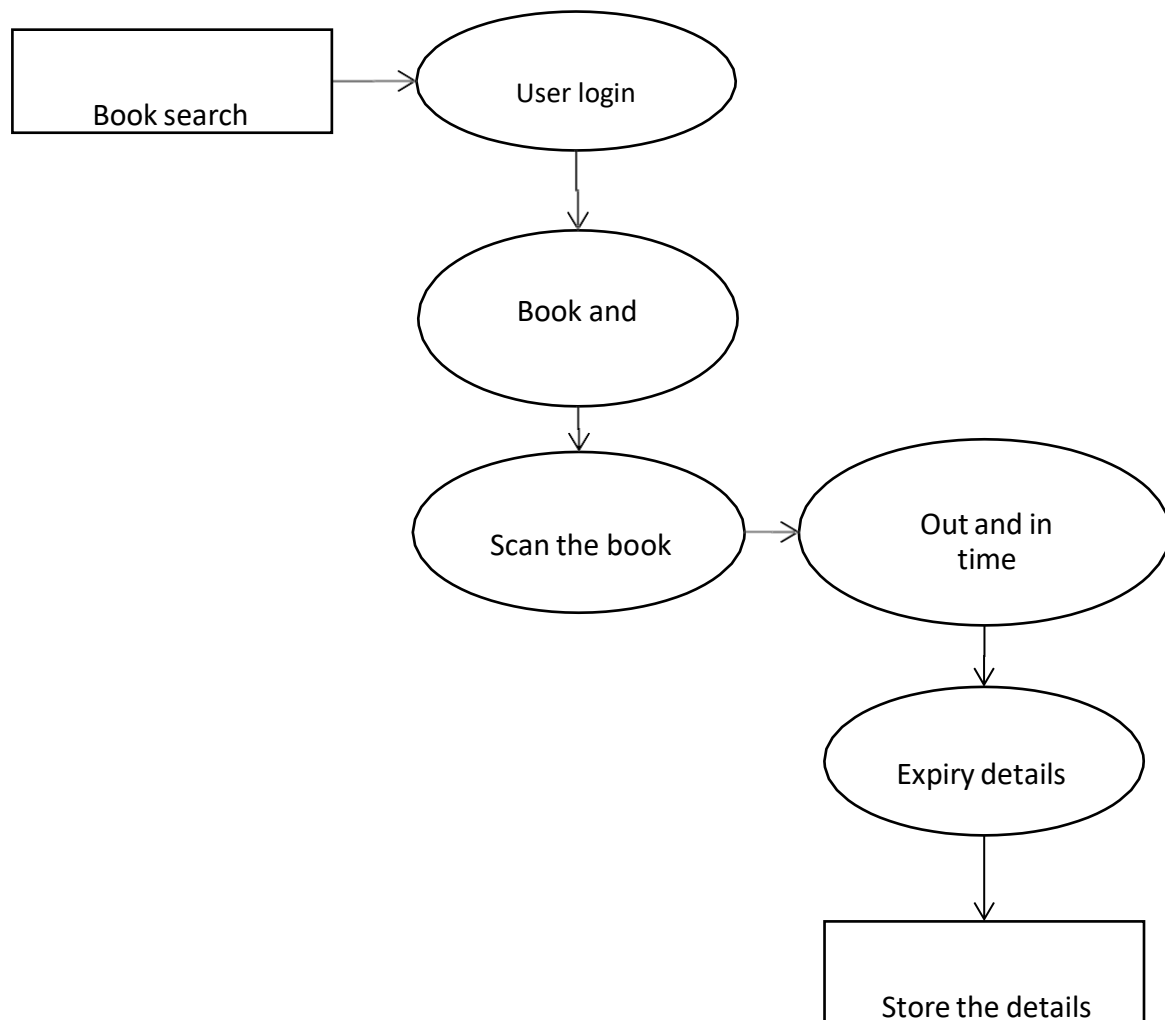
Figure No.5.1.2: Data Flow Diagram (Level 1)

\
LEVEL 2

A Data Flow Diagram (DFD) tracks processes and their data paths within the business or system boundary under investigation. A DFD defines each domain boundary and illustrates the logical movement and transformation of data within the defined boundary. The diagram shows 'what' input data enters the domain, 'what' logical processes the domain applies to that data, and 'what' output data leaves the domain. Essentially, a DFD is a tool for process modeling and one of the oldest.

```
┌──────────────────┐        ╭──────────────╮
│                  │        │   Send SMS   │
│   Alert system   │───────▶│              │
│                  │        ╰──────┬───────╯
└──────────────────┘               │
                                    ▼
                            ╭──────────────╮
                            │  Remainder   │
                            │              │
                            ╰──────┬───────╯
                    ┌──────────────┴──────────────┐
                    ▼                              ▼
           ┌────────────────┐            ┌────────────────┐
           │  Borrow book   │            │  Return book   │
           └────────────────┘            └────────────────┘
```

Figure No.5.1.3: Data Flow Diagram (Level 2)

Solution Architecture

Library Management System


```
                                              ┌─────────────────────────┐
                                              │      checkoutSystem     │
                                              ├─────────────────────────┤
                                              │  checkoutBook() void    │
                                              │  returnBook() void      │
                                              └─────────────────────────┘

                                              ┌─────────────────────────┐
                                              │       searchBooks       │
                                              ├─────────────────────────┤
                                              │  search() void          │
                                              └─────────────────────────┘

        ┌─────────────────────────┐
        │        User  Panel      │
        ├─────────────────────────┤
        │ Initializes checkout/return
        │ Initializes book search  │           ┌─────────────────────────┐
        │ Log out                  │           │         Logout          │
        └─────────────────────────┘           ├─────────────────────────┤
                                              │  Initializes logout system
┌─────────────────────────┐                   └─────────────────────────┘
│       Login System      │
├─────────────────────────┤
│ Initializes User  Panel │                   ┌─────────────────────────┐
│ Initializes Admin Panel │                   │       manageGenres      │
│ Register  User          │                   ├─────────────────────────┤
└─────────────────────────┘                   │  addGenre() void        │
                                              │  removeGenre()  void    │
                                              └─────────────────────────┘

        ┌─────────────────────────┐           ┌─────────────────────────┐
        │        Admin Panel      │           │       manageBooks       │
        ├─────────────────────────┤           ├─────────────────────────┤
        │ Initializes add/remove  genres      │  addBook()void          │
        │ Initializes add/remove  books       │  removeBook() void      │
        │ Initializes manage users │          └─────────────────────────┘
        │ Initializes fees         │
        │ Log Out                  │           ┌─────────────────────────┐
        └─────────────────────────┘           │       manageUsers       │
                                              ├─────────────────────────┤
                                              │  viewUsers() String     │
                                              │  removeUser()  void     │
                                              └─────────────────────────┘

                                              ┌─────────────────────────┐
                                              │       manageFees        │
                                              ├─────────────────────────┤
                                              │  assignFee() void       │
                                              └─────────────────────────┘
```
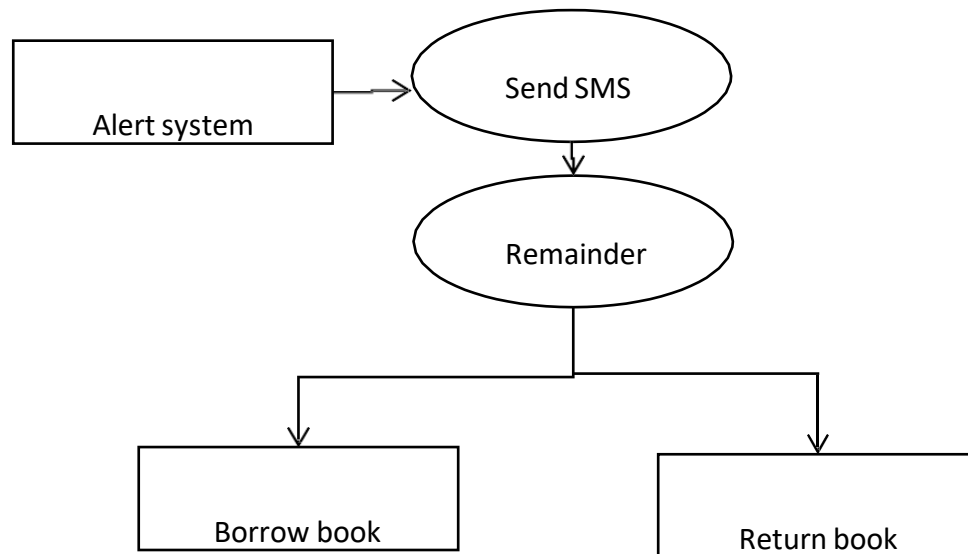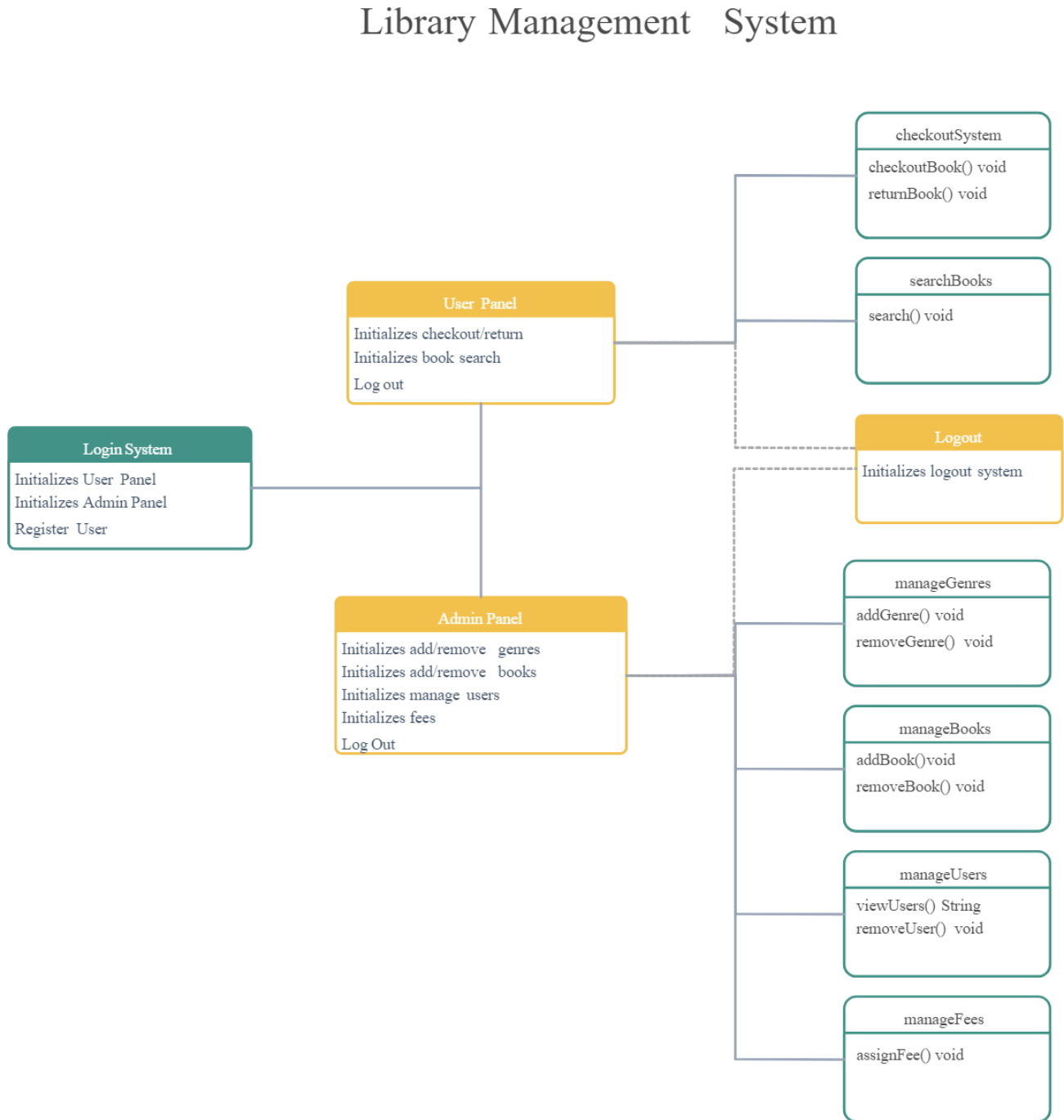
6. PROJECT PLANNING & SCHEDULING

Technical Architecture

① A transaction is requested

② A Block that represents the transaction is created

③ The Block is sent to every node in the network

④ Nodes validate the transaction

Nodes receive a reward for the Proof of Work

⑤ The Block is added to the existing Blockchain

⑥ The transaction is complete

End User ⇄ Frontend (VS code) ⇄ Meta mask ⇄ Solidity code (block chain code)
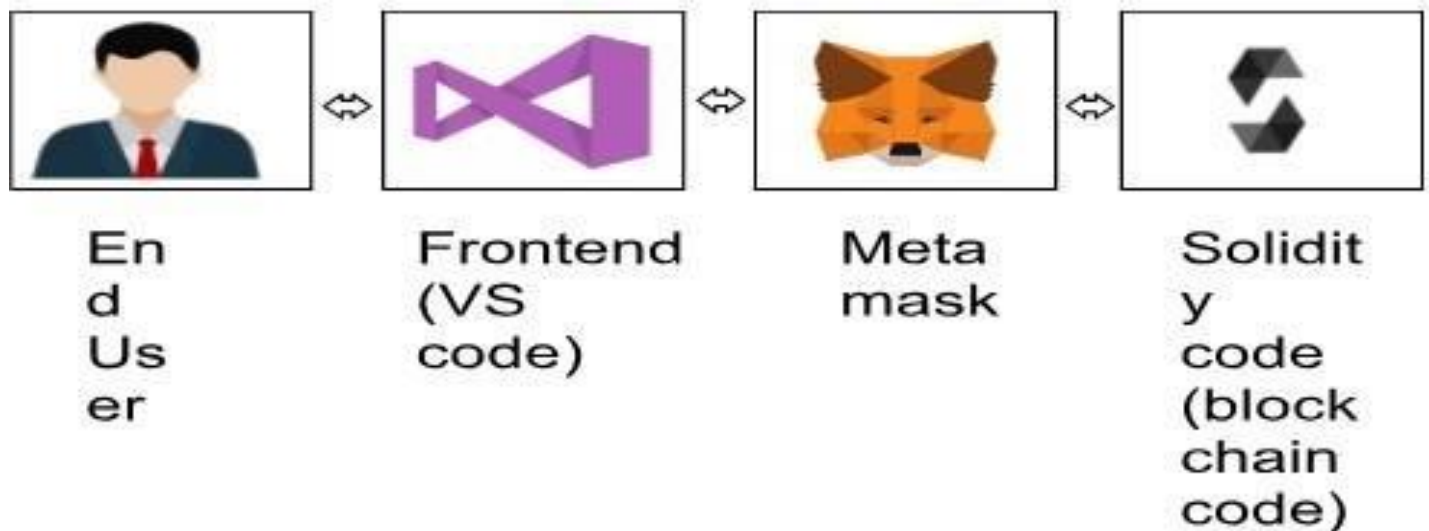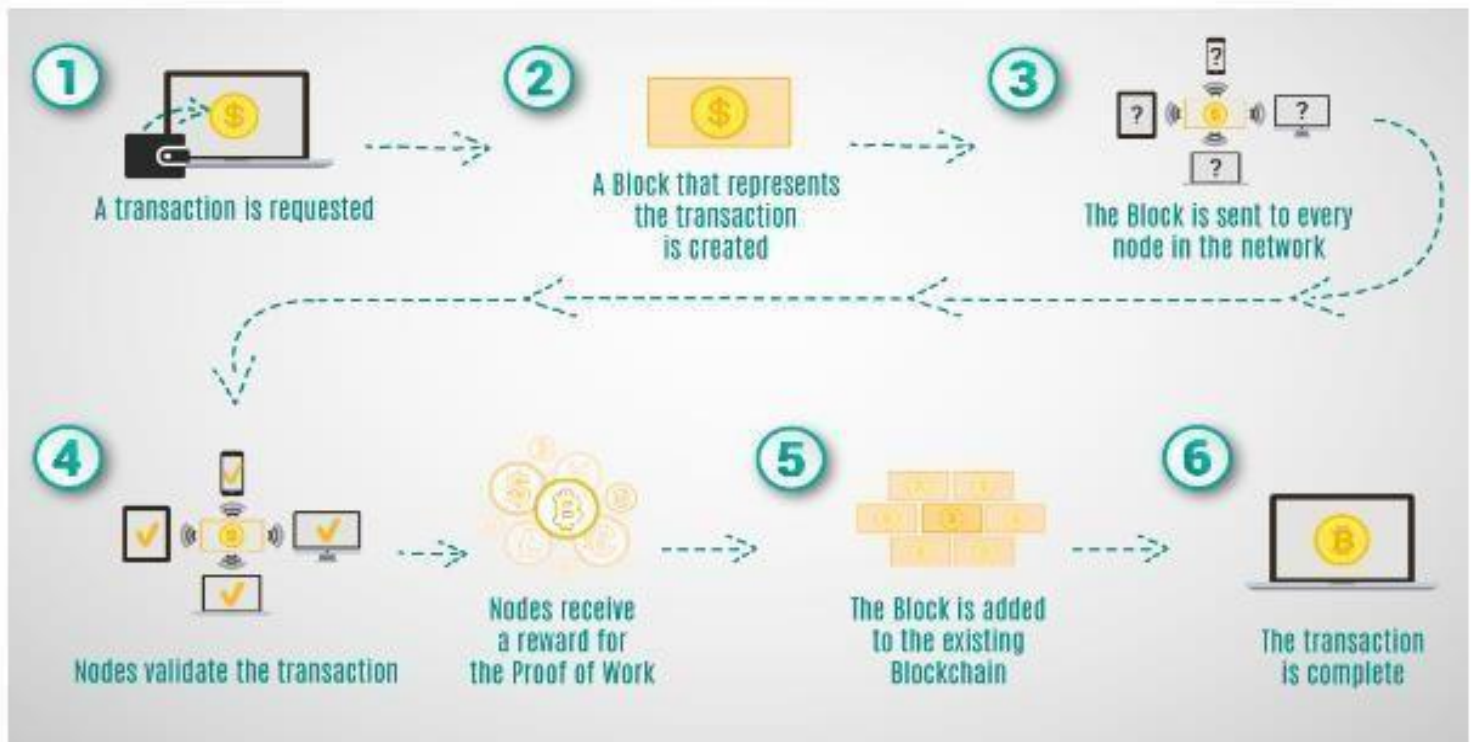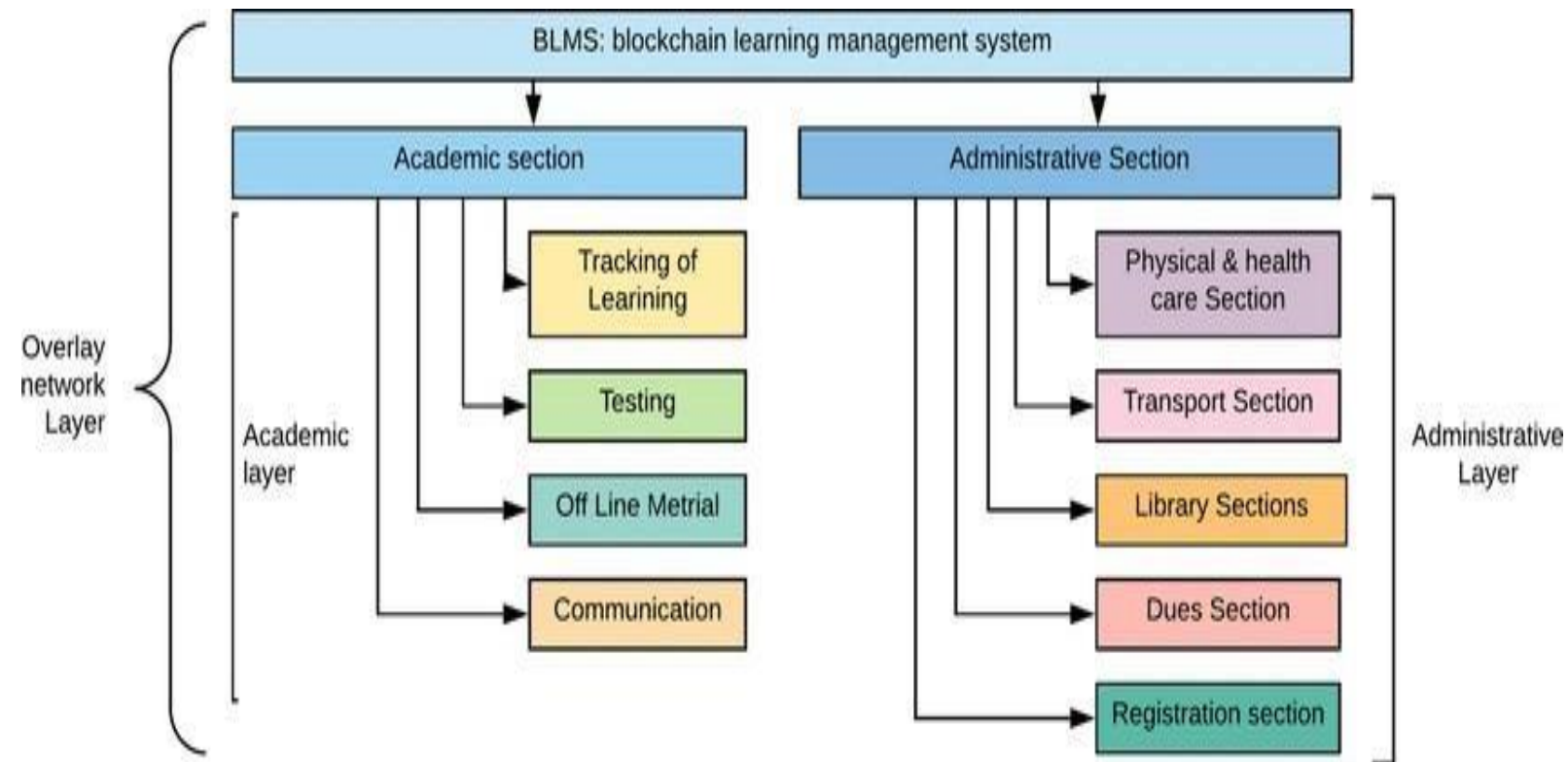
Sprint Planning & Estimation
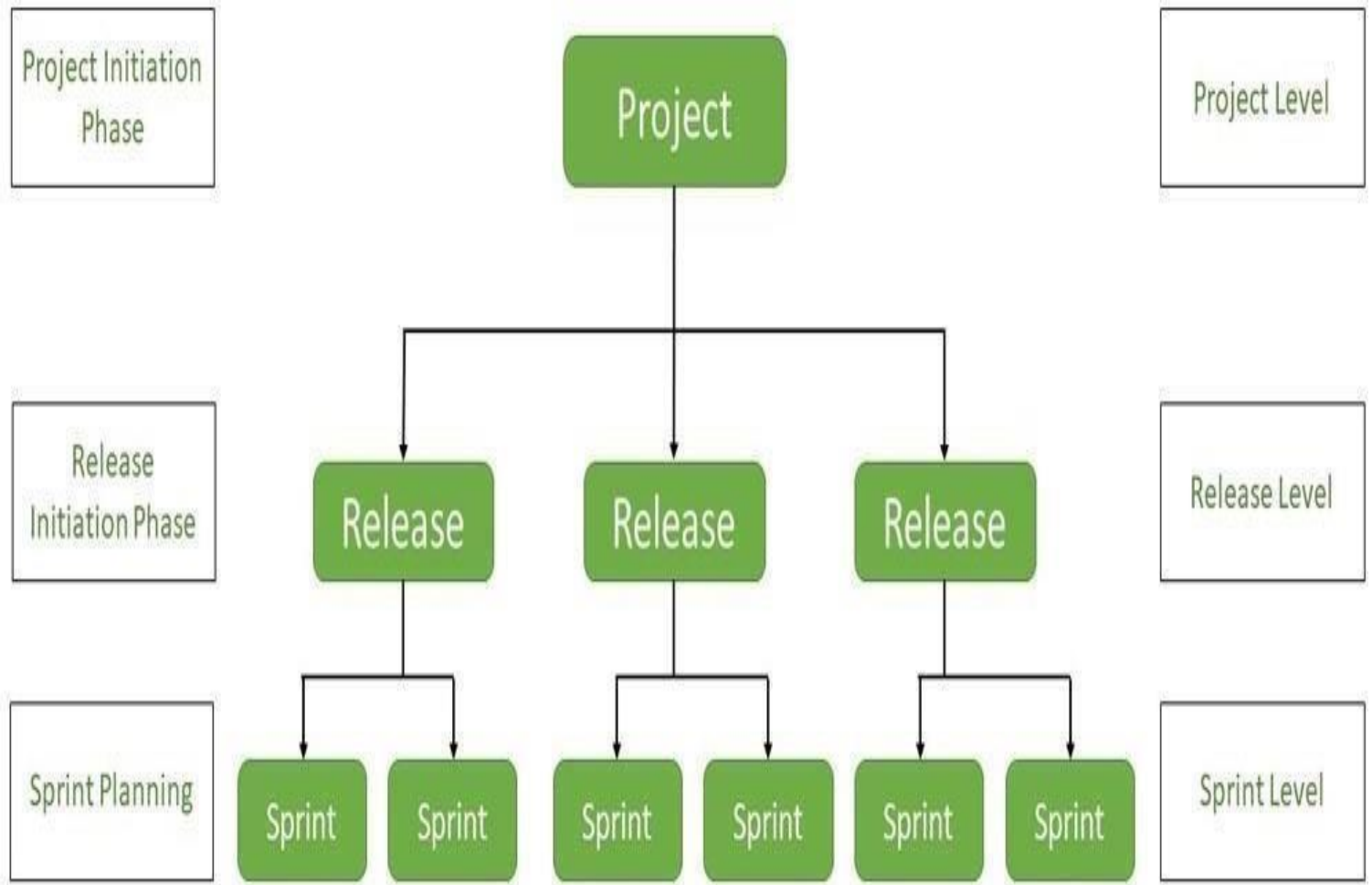
Sprint Delivery Schedule

7. CODING & SOLUTIONING

1. Acquisition management module

Libraries need to keep refreshing their resources but the full acquisition cycle is manually intensive. Keeping track of all the different cogs involved is easier with Acquisition Management Module. IT starts with the selection of resources which is done using pre-order bibliographic searching of the library catalog to avoid duplication and then

- The order is placed

- The goods are received

- Quality checked,

- Invoices processed

- Payment is made to vendors

- Records of the acquisitions are maintained

- Automatic allocation of book IDs to new acquisitions

2. Catalog management module

A very important feature is **catalog management,** the method by which metadata is created representing the knowledge resources of your library such as books, articles, documents, audio clips, maps, digital content. The software will digitally keep track of what is available in the library and catalog the content by title, subject, author and date of publishing.

The system uses **rack numbers** and **location identifiers** to catalog library resources so that students and staff know exactly which shelf of which rack the book or knowledge resource they are looking for can be retrieved from.

3. User management module

A detailed database of users with their name, ID, login and password is created. This helps in keeping track of the member's library usage. Also, a multi-user environment ensures that many users can use the software without speed or access issues.

7.2 Feature 2

1. Online Public Access Catalog (OPAC)

Modern library management systems have web-based OPACs that allow patrons to interface with the library. For example, Koha's OPAC goes beyond merely being a search tool and enables patrons to reserve books, manage their accounts, pay library fees online, track their circulation history, and even make reviews and suggestions for new books.

2. Staff Interface

Web-based library systems have a separate login page for library staff that can be access via web-browser via the internet or through a local network (same as the OPAC). Unlike desktop-based library software that use separate programs for each module, a web-based software like Koha gives you access to all the modules via a single, clean interface.

3. Reports

How do you check which items are the most circulated within a period of time? Or which patrons are reading less books compared to the previous years? This is where a reports module comes in handy. It will help you to keep track of your library and its many activities so that you can keep running a steady and effecient operation.

7.3 Database Schema

1. Book _Details:

This is the master table for all the books that are available in the Library. This table contains the complete list of books that are available in the library. Each Book id provided with a unique ISBN which serves as a primary key. The book details include the ISBN, Book Title, the year in which that particular book was published, the type of binding either soft cover or hard cover and the category.

2. Columns

ISBN: This is unique ID given to every book Since there may be a large no. of books with same TITLE, this ISBN no. will help us to distinguish between books of same title.

Book_ Title: Provides the name of the book.

Publication_ year: Contains the year of publication in 'YY' format (eg:2009à09)

Language: Contains the language in which this book was published.

Category _Type
This column contains the Category ID whose details can be fetched form the category _master table. The category ID is a Unique number given to each category.

3. Binding _Id

This column contains the Binding ID whose details can be fetched form the Binding Detailstable. The Binding ID is a Unique number given to each type Binding.

No _of _Copies _Actual: This column contains the total no. of copies of each book that were initially present.

No _Of _Copies_ Current: This column contains the total no. of copies of each book that were currently available .

4. Binding _Details:

This table is the Master table for the binding types. This includes the binding ID and Binding Name. The Binding ID serves as a primary key.

Columns:

Binding_ ID: This column contains the Unique number that was given to each type of binding.

Binding _Name: This column give the names of different types of binding.

5. Category _Details:

This includes the Category ID and Category Name. The Category ID servers as a primary key. Columns:

Category _ID: This column contains the Unique number that was given to each type of Category.

Category _Name: This column give the names of different types of categories.

6. Borrower _Details:

This table contains the details of all the persons who lent a book from the library. Each Student will be given a Unique borrower ID. All the library related activity for a particular person will be captured based on the Borrower ID. This table will be used to track the borrowing records. The borrower ID will serve as a primary key here.

Columns:

Borrower _ID: Unique ID given to each Student.

Book _ID: This column contains the book ID which was give to the borrower.

Borrowed _From _Date: The date on which the book was given a particular borrower.

Borrowed _To _Date: The date on which that book was supposed to be returned back or should be renewed.

Actual _Return _date: The date on which the borrower returned the book to the library.

Issued _by: The ID of the Librarian who issued book to the borrower.

7. Staff _Details:

This table contains the details of the staff in the Library. Each Staff member will be given a unique User ID which serves as a Primary Key.

Columns

User _ID: The unique ID given to each staff member present in the Library.

User _Name: The Name of the staff member.

Is _Admin: Just checking user is admin or not.

Designation: The role of the staff member in the library such as librarian, assistant, etc.

8. Student _Details:

This table contains the details of all the students they are eligible for availing Library facilities. Each student will be provided with a unique Student ID and Borrower ID. The student ID will be Primary Key, whereas Borrower _ID and Phone _no will be Unique.

Columns:

Student _id: Unique ID given to Each Student.

Student _Name: The Name of the Student.

Sex : Gender of the Student either Male or Female.
Date _Of _Birth: The Date of Birth of the student.

Borrower _ID: The borrower ID assigned to each student.

8. PERFORMANCE TESTING

8.1 Performace Metrics

Department: This is contains student department.

Contact _Number: Contact number of the student.

9. Shelf _Details:

This table contain the position of the book…That means which floor and shelf the book is situated.

Column:

Shelf _Id: Contains the shelf number.

Floor: Which floor the shelf is situated.

- **Functional Testing**: It is conducted to evaluate the blockchain solution compliance with specific feature requirements as it often describes what the system does. Functional testing is a part of black-box testing, i.e., it only examines the application functionality without looking into its internal workings or structures. So, the QA doesn't have to know the programming language used or how the component of the blockchain system has been implemented to perform functional tests. Integration testing, unit testing, smoke testing, and regression testing are some examples of functional testing.

- **Performance Testing**: It ensures that the blockchain application is working accurately under expected conditions and workloads. Performance testing evaluates the quality and capability of the blockchain solution in terms of scalability, speed, stability, and reliability under changing

conditions. It aims to remove performance bottlenecks and identifies areas of improvement while making the blockchain solution fast, stable, and reliable.

- **Security Testing**: It detects and identifies security vulnerabilities in the blockchain system's network, client-side, or server-side. It aims to protect the system from possible intruders by ensuring that there are no possible loopholes, risks, threats, or vulnerabilities in the system. Confidentiality, Authenticity, Authority, Integrity, Availability, and Non-Repudiation are the core principles followed in security testing. Simply put, it is used to debug the system, allowing blockchain solutions to perform highly secured transactions.

- **Node Testing**: Every diverted node available on the network should be tested independently for accurate recording and a risk-free connection.

- **Penetration Testing**: It is a modeled cyber attack on one's blockchain system that helps in recognizing any exploitable vulnerabilities. Penetration testing consists of attempted breaches on servers, APIs, etc. to discover weaknesses prone to injection attacks on the code.
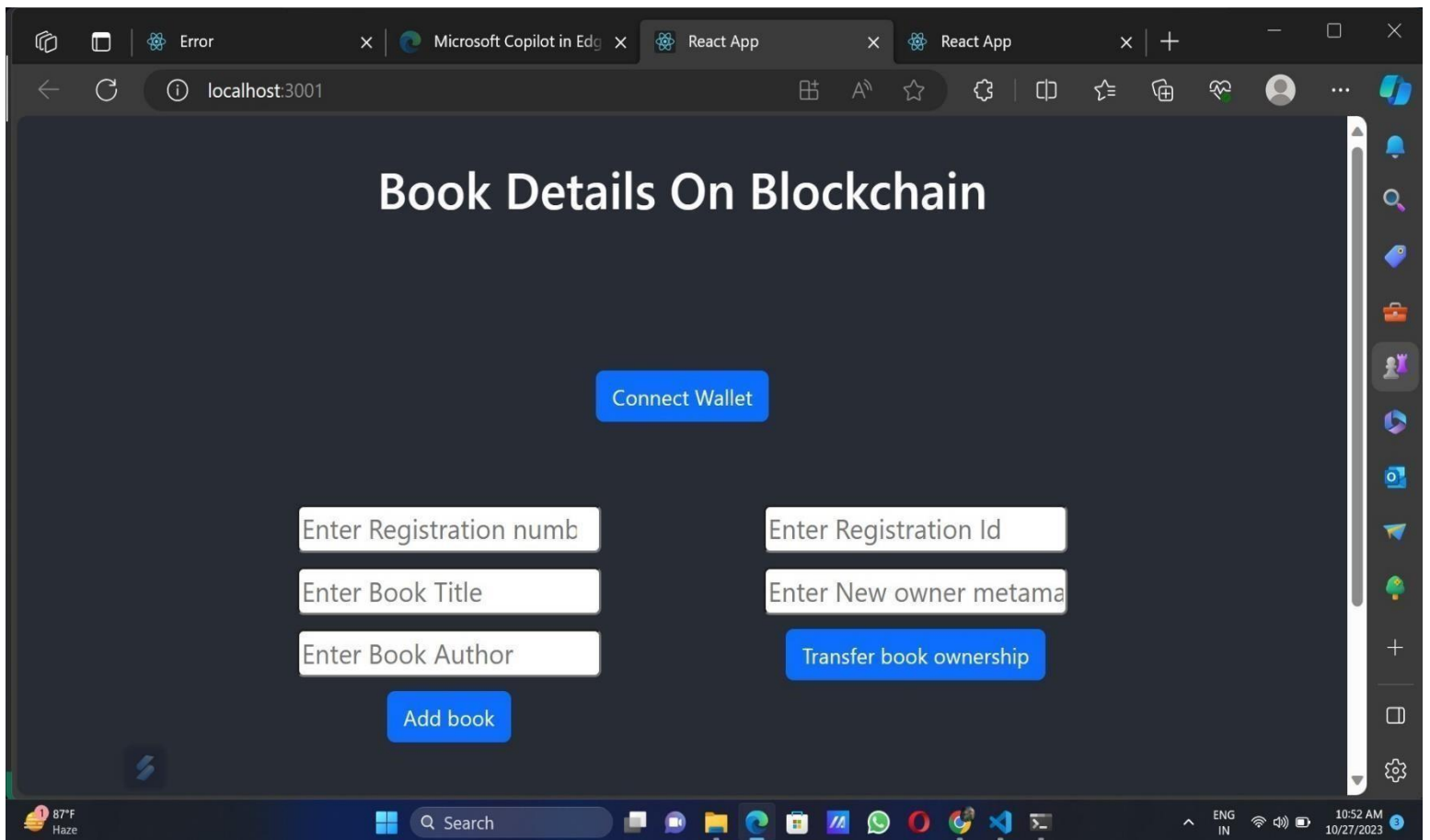
9. RESULTS

9.1 Output Screenshot

Figure: Webpage of the Library Management System

1. Centralized Management

10. ADVANTAGES & DISADVANTAGES

Advantages:

Koha provides a centralized platform for managing all aspects of a library. From book acquisition to circulation, everything can be managed from a single location. As a result, library management becomes easier and errors are less likely to occur.

2. Improved Access

Koha enables patrons to access the library's resources from anywhere in the world. This is made possible through the system's web-based interface, which allows patrons to browse the library's catalog, place holds, and renew items online.

3. Easy Cataloging

Koha simplifies the cataloging process by allowing librarians to import catalog records from external sources. This saves time and reduces the chances of errors.

4. Efficient Circulation

Koha streamlines the circulation process by automating tasks such as check-in, check-out, and hold management. Consequently, patrons will have a better experience and wait times will be reduced.

5. Real-time Reporting

Koha provides real-time reports on library usage, which helps librarians make informed decisions. Reports on circulation statistics, patron behavior, and resource usage are readily available.

6. Multi-Lingual Support

Koha supports more than 100 languages, including English, Chinese, German, Japanese, Italian, and more, making it accessible to libraries in different parts of the world. Libraries that serve multicultural communities will particularly benefit from this feature.

7. Interoperability

Koha is designed to work with other software applications, such as digital repositories and content management systems. In this way, Koha can easily be integrated with existing library systems.

8. Cost-effective

Koha is open-source software, which means it is free to use and can be customized to meet the specific needs of a library. This makes it an affordable alternative to commercial library management systems. This is one of the biggest advantages of the library management system.

9. Community Support

Koha has a large and active community of users and developers who provide support, share knowledge, and contribute to the development of the system. This community-driven approach ensures that Koha is constantly evolving and improving.

10. Security

Koha is a secure system that is regularly updated to address security vulnerabilities. This ensures that the library's data is protected from unauthorized access.

Disadvantages:

1. **Copyright:** One notable disadvantage of digital libraries is the complex issue of copyright. Digital resources often come with copyright restrictions and licensing agreements that dictate how users can access, use, and share the materials. Navigating copyright laws and ensuring compliance can be challenging for both library administrators and users, potentially limiting the availability of certain resources or imposing restrictions on their use.

2. **Speed of Access:** While digital libraries offer the convenience of instant access to resources, the speed of access can be hindered by various factors. Slow internet connections, server congestion, or high demand during peak times can result in frustrating delays when retrieving or downloading materials. This can impede the seamless and efficient retrieval of information,

particularly for users with limited internet bandwidth or in areas with unreliable internet infrastructure.

3. **Initial Cost is High:** Establishing a digital library requires significant upfront investment in hardware, software, digital preservation systems, and staff training. The cost of digitizing print materials, creating metadata, and developing user interfaces can be substantial. These initial costs may pose a challenge for smaller institutions or organizations with limited financial resources, hindering their ability to establish comprehensive digital library collections.

4. **Bandwidth:** Bandwidth limitations can pose a significant challenge when accessing digital library resources, particularly for users in regions with limited internet connectivity or in areas where bandwidth is restricted. Large files, multimedia content, or high-resolution materials can consume considerable bandwidth, leading to slow loading times, buffering issues, or even complete inaccessibility for users with low bandwidth connections.

5. **Efficiency:** While digital libraries offer numerous advantages in terms of information retrieval, the sheer volume of resources available can also be overwhelming. Users may face challenges in navigating complex search interfaces or dealing with irrelevant search results. The efficiency of digital library systems in organizing and categorizing resources, as well as the effectiveness of search algorithms, can impact the overall user experience and the ability to find relevant and accurate information quickly.

6. **Security:** Digital libraries face security challenges that must be addressed to protect sensitive information and maintain user trust. Cyber threats such as hacking, data breaches, and unauthorized access pose risks to the integrity and confidentiality of digital library resources. Libraries must implement robust security measures, including encryption protocols, firewalls, user authentication systems, and regular security audits to safeguard against potential vulnerabilities. Ensuring data privacy and protection is essential to maintain the trust of users and to safeguard intellectual property rights within the digital library environment.

7. **Environment Issue:** Although digital libraries are often perceived as more environmentally friendly than traditional print libraries due to reduced paper usage, their environmental impact should be considered holistically. The significant energy requirements of data centers, server maintenance, and the constant need for technology upgrades contribute to the carbon footprint

of digital libraries. Additionally, the disposal of outdated electronic devices and the proper management of electronic waste pose environmental challenges that need to be addressed

11. CONCLUSION

The emergence of new technology has significantly increased the challenges facing libraries. Emerging technologies like BAR codes in the modern, difficult environment need the sharing of information handling for varied library users. The platform for the Library database is Python Qt5. The addition of books and user IDs to the database is done by the librarian using BAR codes.

Currently, the majority of students and faculty members utilize smart phones, making it easier to scan, search, explore, and make decisions about BAR codes. Books, which reduces the time it takes the library to issue books. Both user and librarian efforts will be reduced as a result. The problems, which existed in the earlier system, have been removed to a large extent. And it is expected that this project will go a long way in satisfying user's requirements. The computerization of the Library Management will not only improves the efficiency but will also reduce human stress thereby indirectly improving human recourses.

12. FUTURE SCOPE

The efficient utilization of the technology also depends upon the information to be written in tag. These applications can lead to significant savings in labor costs, enhance customer service, lower book theft and provide a constant record update of new collections of books. In future, we can extendthe framework to implement the framework in real time android applications.

13. APPENDIX

Source Code

```solidity
// SPDX-License-Identifier: MIT pragma
solidity ^0.8.0;

contract Book Registry { address
public owner;

constructor() { owner
= msg.sender;
}

modifier onlyOwner() { require(msg.sender == owner, "Only the owner
can perform this action"); _;
}

struct Book { string
title; string author;
address currentOwner;
}
```

```solidity
mapping(uint256 => Book) public books; uint256
public bookCount;

event BookAdded(uint256 indexed bookId, string title, string author, address indexed
owner); event OwnershipTransferred(uint256 indexed bookId, address indexed
previousOwner, address indexed newOwner);

function addBook(uint256 registration, string memory _title, string memory _author)
external onlyOwner {

books[registration] = Book(_title, _author, owner);
bookCount++; emit BookAdded(registration, _title,
_author, owner); }

function transferOwnership(uint256 registrationId, address _newOwner) external {
require(_newOwner != address(0), "Invalid address"); require(_newOwner !=
books[registrationId].currentOwner, "The new owner is the same as the current owner");
require(msg.sender == books[registrationId].currentOwner, "Only the current owner can
transfer ownership");

address         previousOwner         =         books[registrationId].currentOwner;
books[registrationId].currentOwner = _newOwner;

emit OwnershipTransferred(registrationId, previousOwner, _newOwner); }
```

function getBookDetails(uint256 registrationId) external view returns (string memory, string memory, address) {

Book memory book = books[registrationId]; return
(book.title, book.author, book.currentOwner);

}
}

Connector.js

```javascript
const { ethers } = require("ethers");

const abi = [
 {
  "inputs": [],
  "stateMutability": "nonpayable",
```

"type": "constructor"

"anonymous": false

"inputs": [
 {
  "indexed": true,
  "internalType": "uint256",
  "name": "bookId",
  "type": "uint256"
 },
 {
  "indexed": false,
  "internalType": "string",
  "name": "title",
  "type": "string"

```json
},
{
  "indexed": false,
  "internalType": "string",
  "name": "author",
  "type": "string"


  "indexed": true
  "internalType": "address"
  "name": "owner"
  "type": "address"
}
],
"name": "BookAdded",
"type": "event"
```

```json
    },
    {
      "anonymous": false,
      "inputs": [
        {
          "indexed": true,
          "internalType": "uint256",
          "name": "bookId",
          "type": "uint256"
        },
        {
          "indexed": true,
          "internalType": "address",
          "name": "previousOwner",
          "type": "address"
```

"indexed": true

"internalType": "address"

"name": "newOwner"

"type": "address"
    }
  ],
  "name": "OwnershipTransferred",
  "type": "event"
},
{
"inputs": [
  {
   "internalType": "uint256",
   "name": "registration",
   "type": "uint256"

```json
        },
        {
          "internalType": "string",
          "name": "_title",
          "type": "string"
```

```json
          "internalType": "string"
          "name": "_author"
          "type": "string"
```

```json
      "name": "addBook",
      "outputs": [],
      "stateMutability": "nonpayable",
      "type": "function"
```

```
},
{
 "inputs": [],
 "name": "bookCount",
 "outputs": [
  {
   "internalType": "uint256",
   "name": "",
   "type": "uint256"
  }
 ],
 "stateMutability": "view",
 "type": "function"
```

"inputs":

```json
      "internalType": "uint256"
      "name": ""
      "type": "uint256"
    }
  ],
  "name": "books",
  "outputs": [
    {
      "internalType": "string",
      "name": "title",
      "type": "string"
    },
    {
      "internalType": "string",
      "name": "author",
```

```json
      "type": "string"
    },
    {
      "internalType": "address",
      "name": "currentOwner",
      "type": "address"

    "stateMutability": "view"
    "type": "function"


"inputs": [
  {
    "internalType": "uint256",
    "name": "registrationId",
```

```json
      "type": "uint256"
    }
  ],
  "name": "getBookDetails",
  "outputs": [
    {
      "internalType": "string",
      "name": "",
      "type": "string"
    },
    {
      "internalType": "string",
      "name": "",
      "type": "string"
```

```json
      "internalType": "address"
      "name": ""
      "type": "address"

    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [],
    "name": "owner",
    "outputs": [
      {
        "internalType": "address",
        "name": "",
        "type": "address"
```
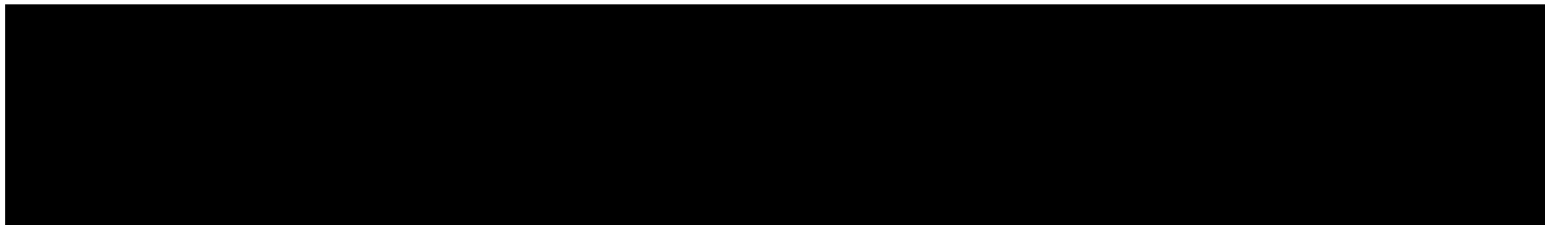
```
 }
],
"stateMutability": "view",
"type": "function"
},
```

```
"inputs":
```

```
"internalType": "uint256"
"name": "registrationId"
"type": "uint256"
```

```
export const signer = provider.getSigner()   export const address =
"0xd8b934580fcE35a11B58C6D73aDeE468a2833fa8"
```

```json
  {
    "internalType": "address",
    "name": "_newOwner",
    "type": "address"
   }
 ],
 "name": "transferOwnership",
 "outputs": [],
 "stateMutability": "nonpayable",
 "type": "function"
 }
]
```

```javascript
if (!window.ethereum) {   alert('Meta Mask Not
Found')
window.open("https://metamask.io/download/")
}
```

```javascript
export const provider = new ethers.providers.Web3Provider(window.ethereum);
```

```javascript
export const contract = new ethers.Contract(address, abi, signer)
```

GitHub & Project Demo link:

Demo link:
https://drive.google.com/file/d/1hvVeJQhbJ4zFcuEwPzZSd4CFDOYXznqC/view?usp=drive_link

GitHub link:

https://github.com/Anandhmech/Power-library-management