

PROJECT REPORT

Date	28 November 2023
Team ID	NM2023TMID04061
Project Name	Blockchain Powered Library Management

Blockchain Powered Library Management

1. INTRODUCTION:

1.1 Project Overview

"Blockchain-Powered Library Management" revolutionizes 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 decentralized environment. Libraries, historical repositories of knowledge, can now seamlessly transition to a digital age with immutable and transparent book records stored on the blockchain.

1.2 Purpose

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 authorized personnel can efficiently add new books or transfer ownership with a single, secure transaction.

2. LITERATURE SURVEY:

2.1 Existing problem

The adoption of Blockchain-Powered Library Management systems brings significant social and business impacts. From a social perspective, it enhances data security, protecting patron information while promoting transparency and trust in library operations. It also streamlines resource sharing and accessibility, making knowledge more readily available to diverse user groups. On the business front, these systems improve administrative efficiency, reducing costs associated with manual cataloguing, late fee management, and resource tracking.

2.2 Reference:

- Verma, Manish. "Amalgamation of Blockchain Technology and Knowledge Management System to fetch an enhanced system in Library", in IJIRT | Vol. 7, Issue 11, April 2021 (pp.474-477)
- Casino, Fran, Thomas K. Dasaklis, and Constantinos Patsakis. "A systematic literature review of blockchain-based applications: current status, classification and open issues." *Telematics and Informatics* 36 (2019): 55-81.
- Jin, T., Zhang, X., Liu, Y. and Lei, K. (2017), "BlockNDN: a bitcoin blockchain decentralized system over named data networking", 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), IEEE, 75-80.
- Frederick, D.E. (2019), "Blockchain, libraries and the data deluge", *Library Hi Tech News*, Vol. 36 No. 10, pp. 1-7.
- Ahram, T., Sargolzaei, A., Sargolzaei, S., Daniels, J. and Amaba, B. (2017), "Blockchain technology innovations", 2017 IEEE Technology & Engineering Management Conference (TEMSCON), IEEE, 137-141.

2.3 Problem Statement Definition:

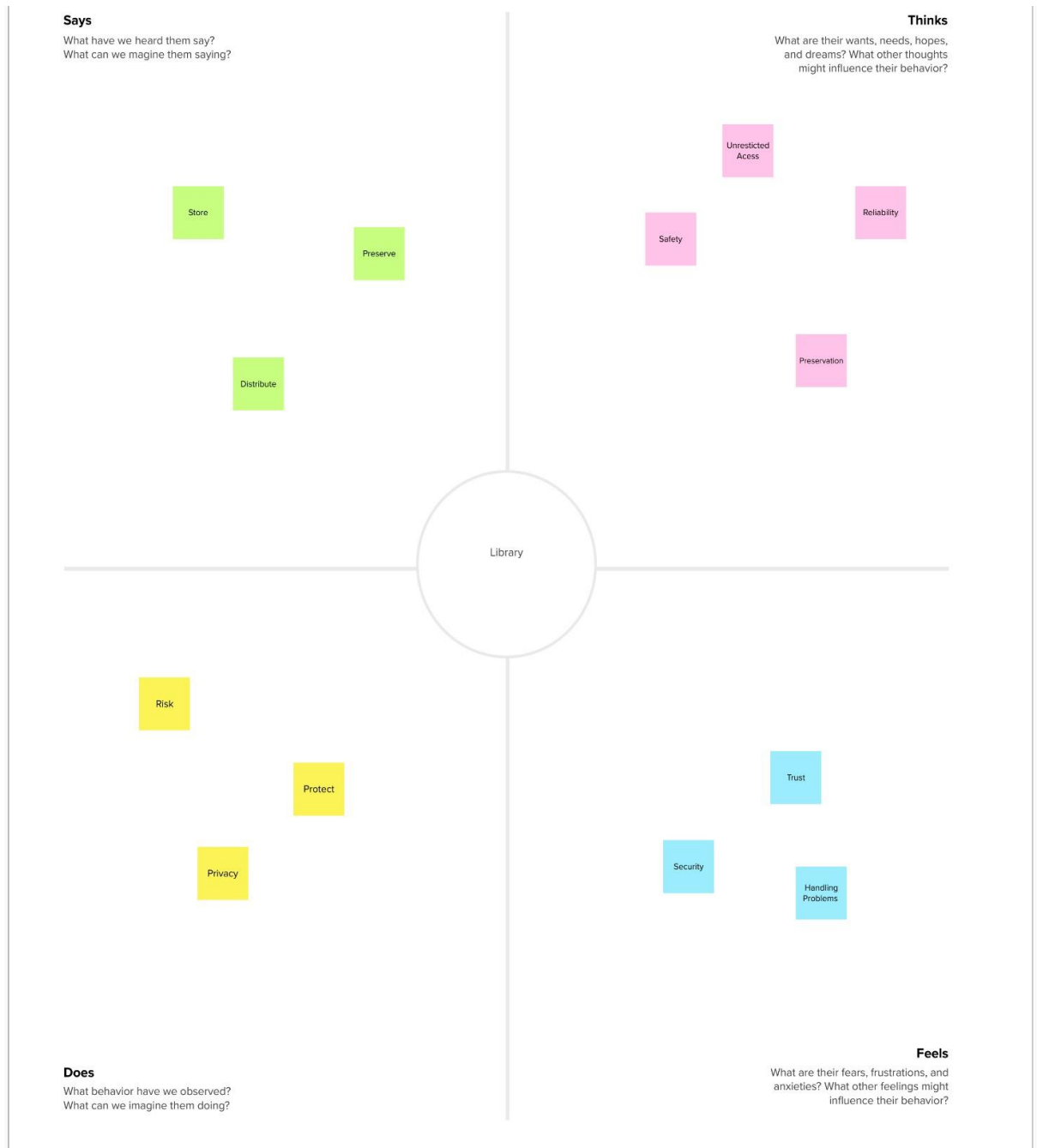
Adequate fund should be made available to libraries by their parent institutions for the implementation of blockchain technology for sustainable library and information practice. In addition, library and information professionals should 'brace for impact' through concerted training and retraining on blockchain, because of the massive changes this tech would create in the library ecosystem.

Capacity building. Library and information professionals should be trained and equipped with skills and technical know-how to work with blockchain technology in order to gain mastery of the tech for easy usability and efficient service delivery.

Library and information professionals should sensitize their user community and the general public appropriately on the ethical use of blockchain as a genuine means of storing, verifying and managing information in order to increase public trust of the technology in libraries. They should also be discouraged from using the tech for malicious purposes as there are grave consequences for doing such.

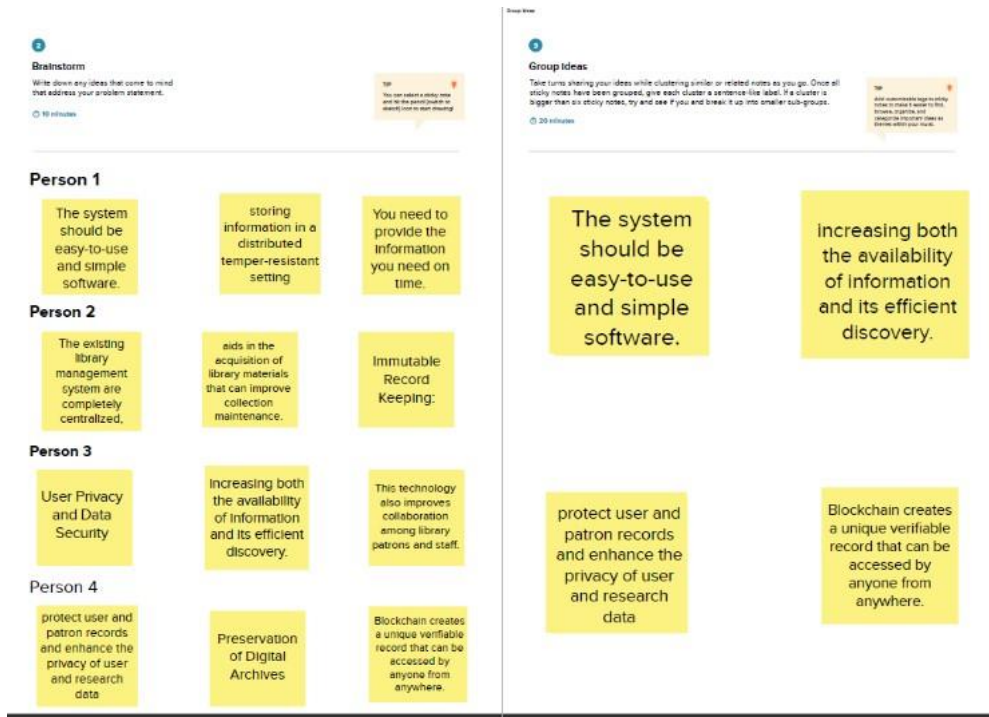
2.IDEATION AND PROPOSED SOLUTION:

2.4 Empathy map canvas:

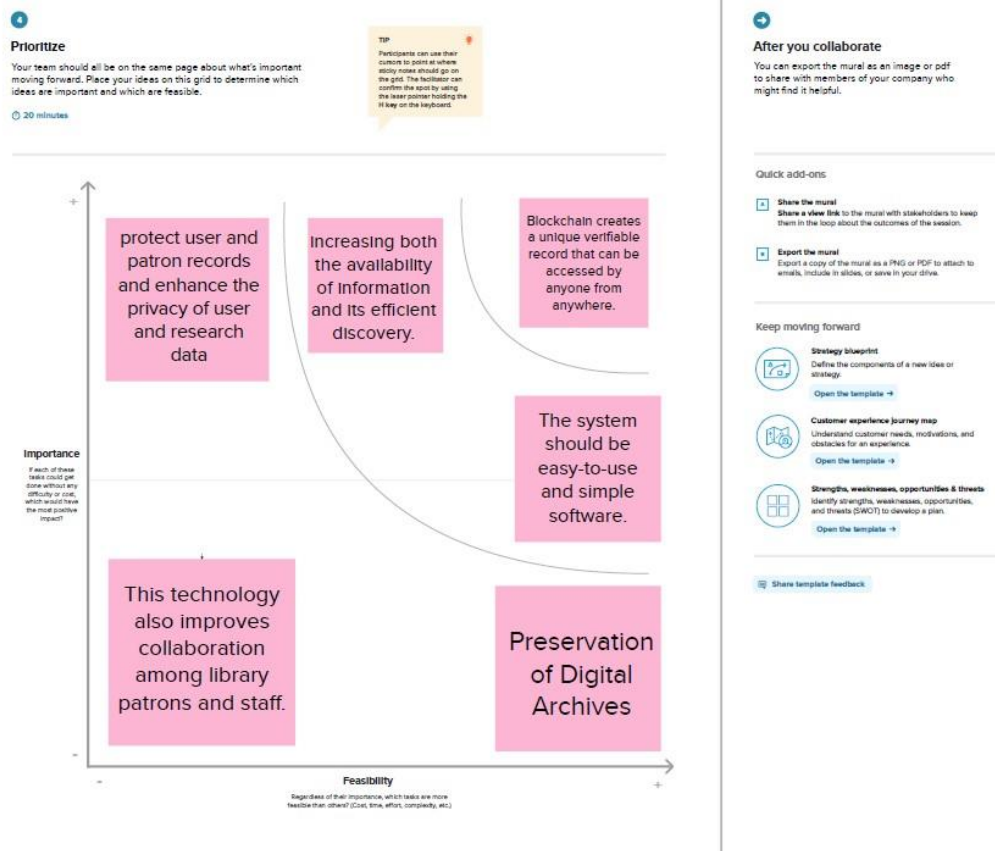


2.5 Ideation and Brainstorming:

Step 1: Team gathering, Collaboration and select the problem statement



Step 3: Idea Prioritisation



4.REQUIREMENT ANALYSIS

4.1 Functional Requirement:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Website	A network of libraries
FR-2	Storing data	Databases
FR-3	Blockchain	Miners
FR-4	Operating System	Servers
FR-5	Processor	2.20 GHz Dual-Core CPU Processor
FR-6	Browser	A compatible browser, for accessing the online module

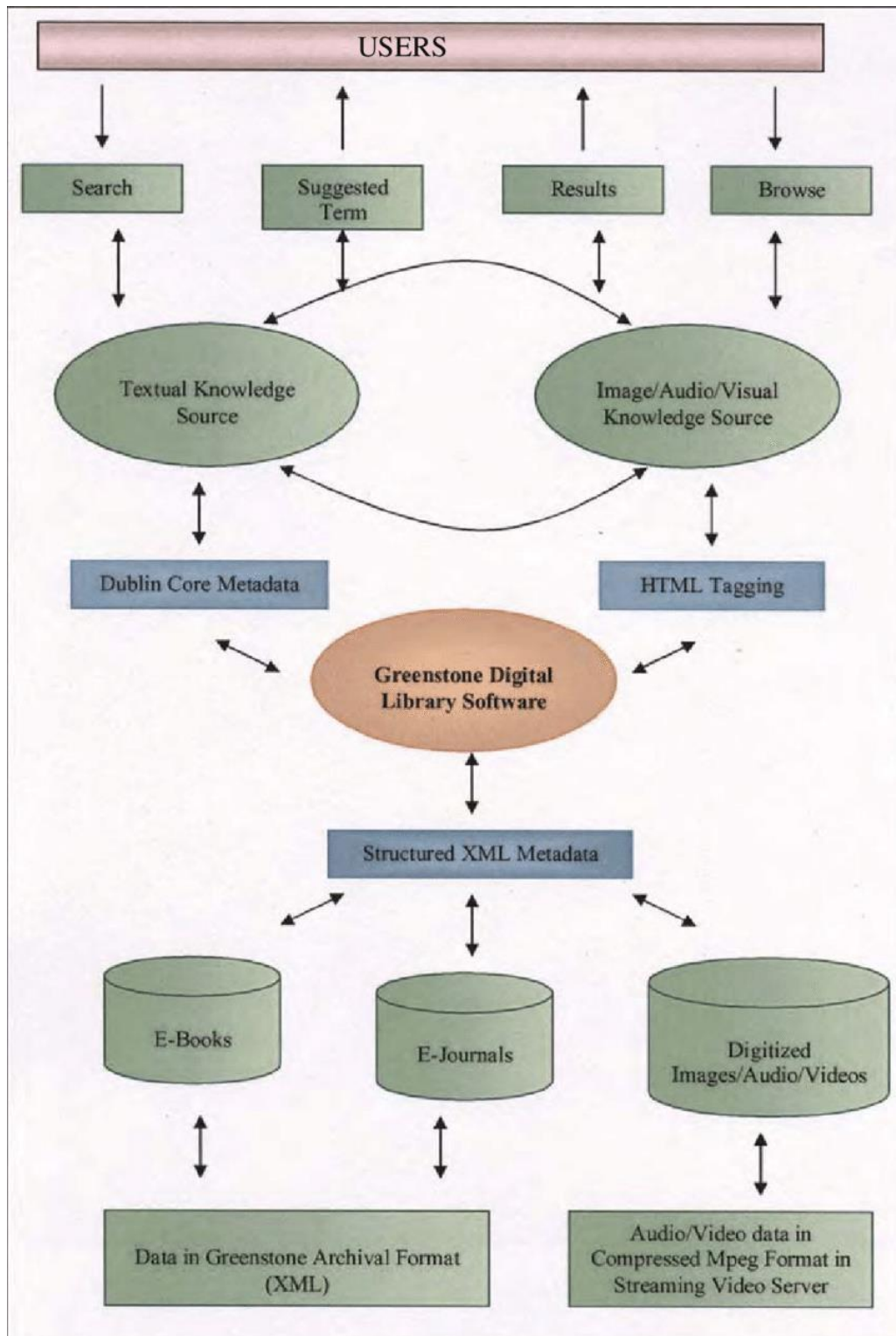
4.2 Non-Functional Requirement:

FR No.	Non-Functional Requirement	Description
NFR-1	Faster settlement	Blockchain technology relies on faster speeds and saves time for institutions and consumers.
NFR-2	Decentralization	Blockchain technology offers a decentralized system that stores the assets in a network and can be accessed via the internet.
NFR-3	Increased capacity	Blockchain technology can increase the capacity of an entire network.
NFR-4	Host Digital Peer-To-Peer Sharing	Library facilitation of peer-to-peer sharing on the far side simply books through blockchain technology
NFR-5	Immutability:	Blockchain uses immutable ledgers, and all databases require trust of a third party to keep them secure from hackers.
NFR-6	Problem solutions:	Before implementing blockchain technology, librarians might keep the solution to each problem

5.PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories:

Data Flow Diagram:

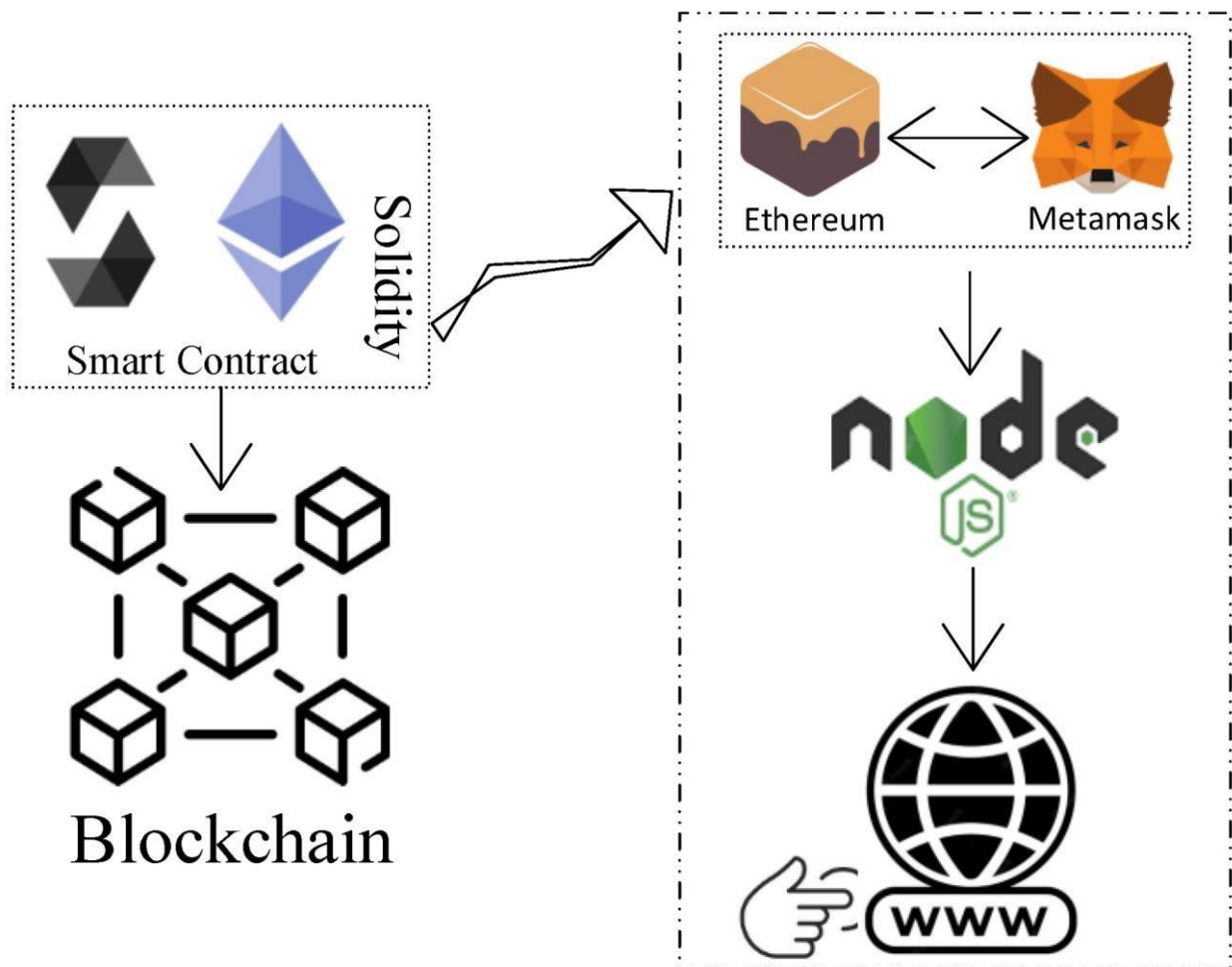
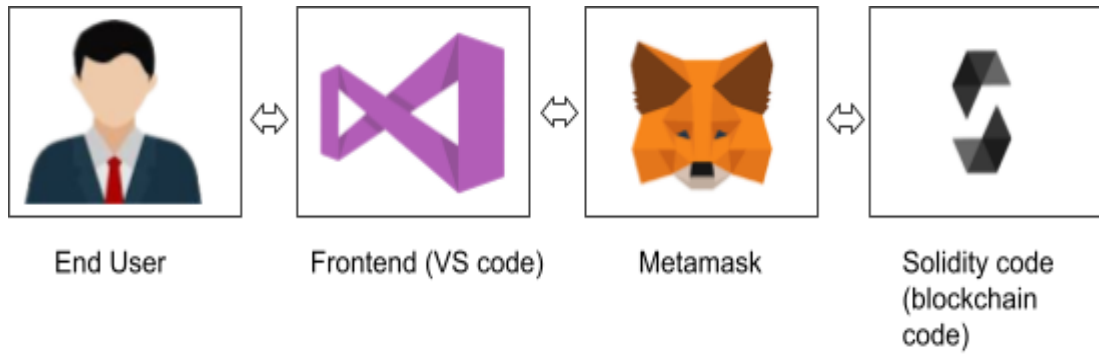


User Stories:

User Story no:	Stories	Priority
US-1	Increased capacity: Blockchain technology can increase the capacity of an entire network.	High

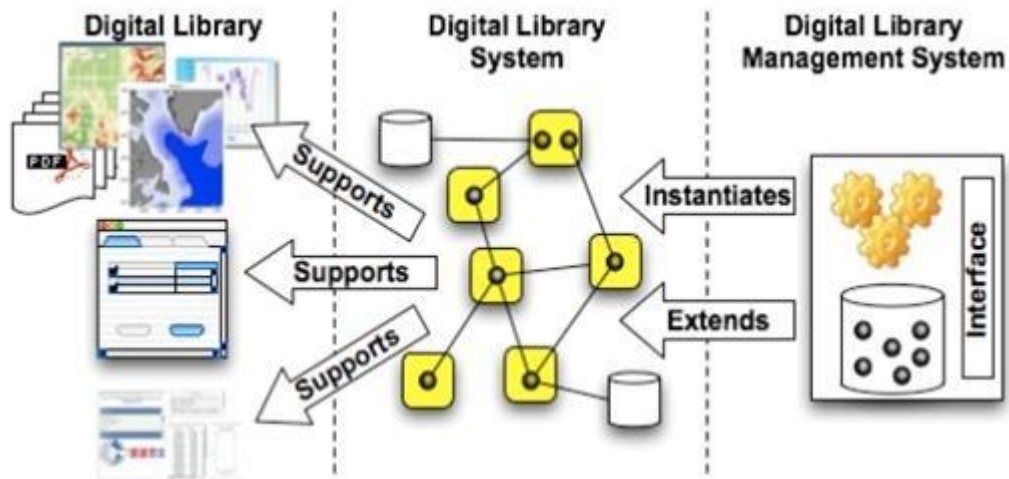
US-2	Project management: There are different factors which should be kept in mind while launching blockchain projects such as programmers, project management team and visionary staff from your libraries.	Moderate
US-3	Immutability: Blockchain uses immutable ledgers, and all databases require trust of a third party to keep them secure from hackers. Blockchain applications, such as Bitcoin, maintain the ledger in a never-ending state of forward momentum	Low
US-4	Better security: Blockchain technology offers better security as it provides for a network of numerous computer nodes that can be used for networking transactions.	High

5.2 Solution Architecture:



6.PROJECT PLANNING

6.1 Technical Architecture:



7.CODING & SOLUTIONING

7.1 Coding:

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

```
contract BookRegistry {
    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;    }
```

```
    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;
```

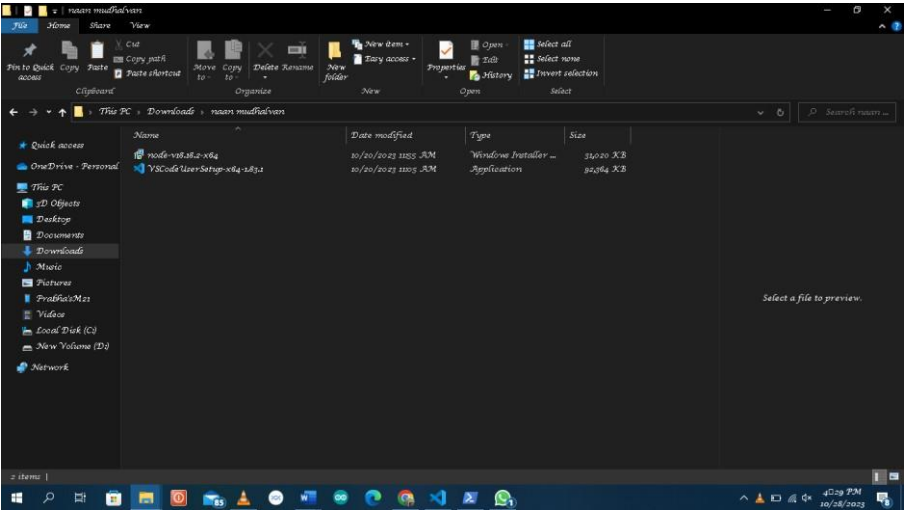
```
        emit OwnershipTransferred(registrationId, previousOwner, _newOwner);
    }
```

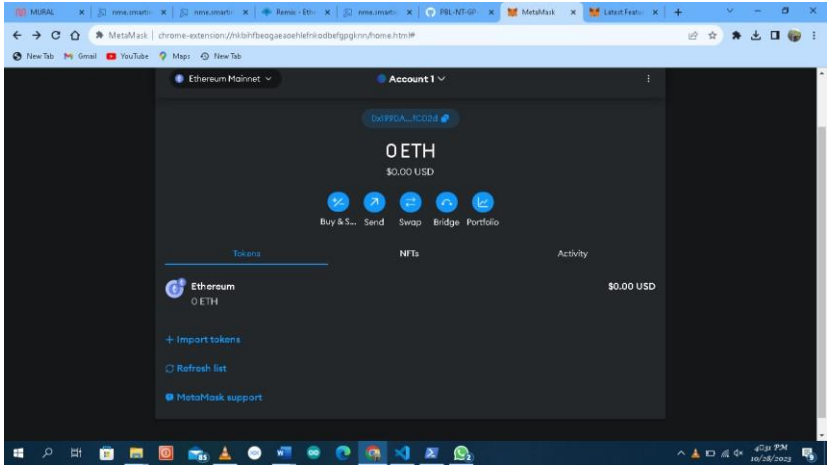
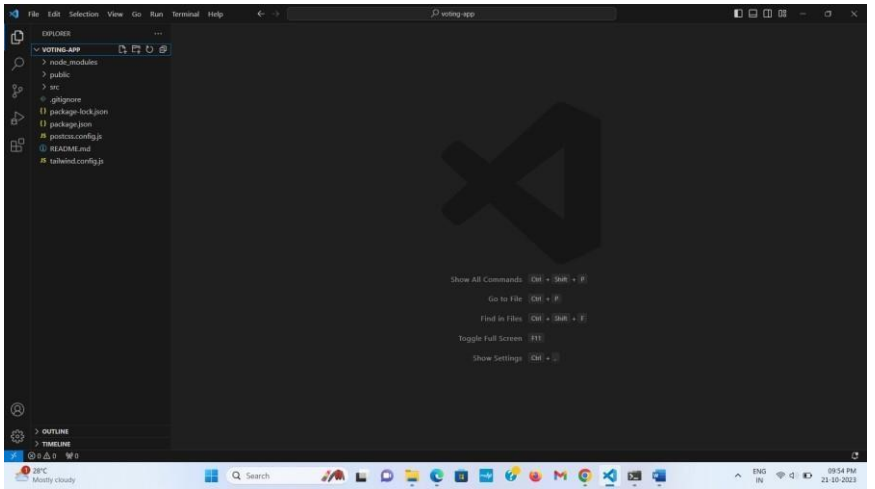
```
    function getBookDetails(uint256 registrationId) external view returns (string memory,
string memory, address) {
```

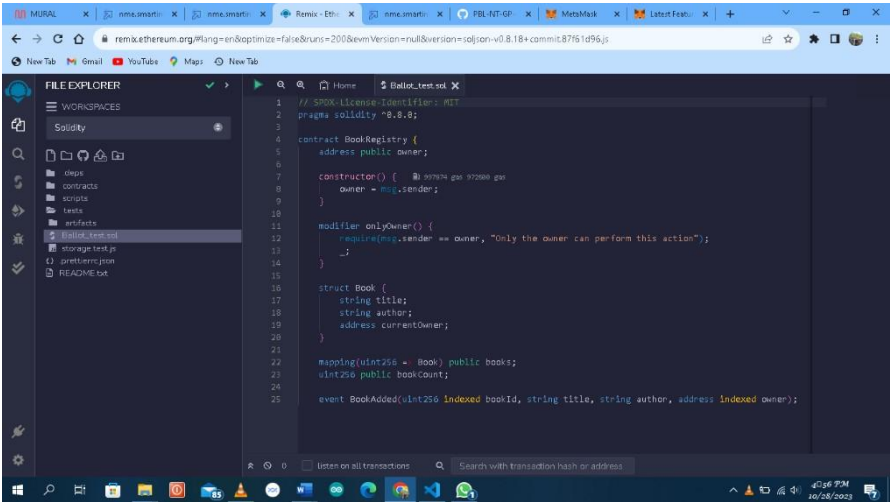
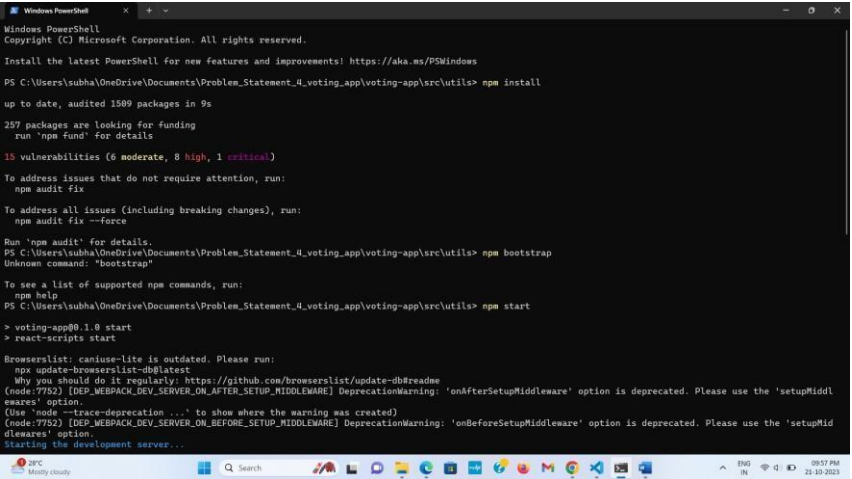
```
        Book memory book = books[registrationId];
return (book.title, book.author, book.currentOwner);
    }
}
```

8.PERFORMANCE TESTING

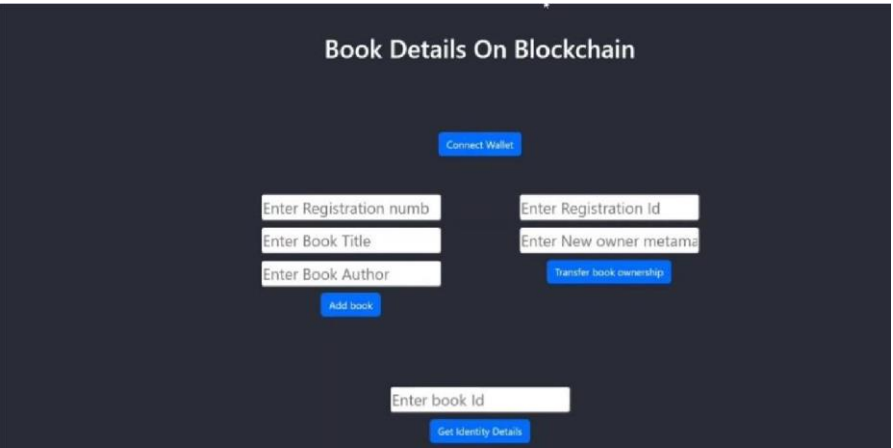
8.1 Performance Metrics:

S.No.	Parameter	Values	Screenshot
1.	Information gathering	Setup all the Prerequisite:	 <p>VS Code & nodejs</p>

			 <p>Metamask</p>
2.	Extract the zip files	Open to vs code	

3.	Remix Ide platform exploring	<p>Deploy the smart contract code</p> <p>Deploy and run the transaction. By selecting the environment - inject the MetaMask.</p>	
4.	Open file explorer	<p>Open the extracted file and click on the folder.</p> <p>Open src, and search for utiles.</p> <p>Open cmd enter commands</p> <p>1.npm install</p> <p>2.npm bootstrap</p> <p>3. npm start</p>	

			
5.	LOCALHOST IP ADDRESS	copy the address and open it to chrome so you can see the front end of your project	

			
--	--	--	--

10.Advantages and disadvantages

Advantages:

1. It creates a decentralized storage system where both the librarians and users can have access to manage the information resources in the library, in tandem with Library 2.0 initiative.
2. A unique identifier code tied to each information resource from the publishers, would help to checkmate copyright infringement.
3. Security, resilience and cost saving. Blockchain is secured and somewhat cheaper to manage the information therein than conventional database management system. Most of the conventional database system demands a special database administrator, who will carry out the database main-tenance and backup schedule, unlike blockchain that is decentralised. Consequently, libraries and information centres will save cost of security and data management by implementing blockchain technology

Disadvantages:

The challenges are having the time and resources to develop use cases which will offer direction to libraries for mistreatment blockchain as a thought tool. Blockchain is associate untested construct in libraries, and there'll still be scepticism and reluctance to pursue its use till there are credible samples of the ways in which blockchain will be used successfully for

library processes. Alternative problems impacting the implementation of blockchain involve standards, privacy, and legalities, and every of those are highlighted within the book.

11.Conclusion

Blockchain-Based Library Management that is depending on the Smart Contract for Library Management and is deployed on JavaScript VM of Remix IDE written in solidity language. Blockchain library management system restricts the use of copyright digital materials with errorless auditing of records of books. Hence, blockchain-based library is the genesis for the next generation Library management system. Thus, blockchainbased library management is Library 2.0 technology.

12.Future scope

The following suggestions were offered for further research:

1. Future research directions should centre on librarians' perception of blockchain technology in light of the increasing hype and criticism of the tech across the globe.
2. Future studies on the application of blockchain technology in libraries and information centres should focus on the acceptance rate of the tech by librarians, with respect to Davis (1989) and Davis et al (1989) Technology Acceptance Models (TAMs).
3. Further researches should also survey the rate of diffusion and adoption of blockchain in libraries and information centres, and analyze the data based on Everett Rogers (1962) Diffusion of Innovations adopter categories. That is, the innovators, early adopters, early majority, late majority and laggards respectively.

13.Appendix

13.1 Source code:

```
// SPDX-License-Identifier: MIT pragma
```

```
solidity ^0.8.0;
```

```
contract BookRegistry {  
    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;
```

```

    }

    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);
    }
}

```

13.2 Github link:

<https://github.com/prabha242/Block-chain.git>

13.3 Project demo link:

<https://drive.google.com/file/d/1Qsc6AUPhQvPFZPNdewFGJ2bDrmjz1zbU/view?usp=sharing>