Blockchain Powered LibraryManagement

A PROJECT REPORT

Submitted by

ANITHA .R

BABY SHALINI. M

RADHA KRISHNA VENI .K

SIVARANJANI .V.P

TEAM ID: NM2023TMID03178

1.INTRODUCTION

PROJECT OVERVIEW

Welcome to the future of library management with Blockchain technology! In this digital era, the traditional library system is evolving, and Blockchain is at the forefront of this transformation. Blockchain Powered Library Management is an innovative solution that leverages the decentralized, transparent, and secure nature of blockchain to revolutionize how libraries operate. This cutting-edge approach ensures the integrity of library records, enhances user privacy, and simplifies the lending and return processes. Let's delve deeper into how blockchain is reshaping the way libraries function in this exciting new chapter of library management.

PURPOSE

Enhanced Security: Blockchain technology provides a highly secure and tamper-proof way of storing and managing library records. It ensures that records cannot be altered or deleted, thus preserving the integrity of the library's data.

Data Integrity: The purpose is to maintain the integrity of library records, ensuring that books and resources are accurately tracked and managed. Users can trust that the information they access is up-to-date and accurate.

User Privacy: Blockchain allows for enhanced user privacy. Users can control their personal information, and sensitive data is stored securely, reducing the risk of data breaches and privacy violations.

2.LITERATURE SURVEY ON BLOCKCHAIN-POWERED LIBRARY MANAGEMENT

S.NO	Title of the paper	Authors	Algorithm	Advantages	Disadvantages
1.	Towards A Blockchain Enabled Integrated Library Management System Using Hyperledger Fabric	Zohaib Ahmad Chugtai, Rizwan Malik, Hammad Raza, Muhammad Saleem	Academic and Public libraries are Considered as free educational resources for the readers.	Capabilities and potential benefits for libraries blockchain is about storing information in a distributed, tamper-resistant setting	Blockchain is not a Distributed computing system
2.	The Study on BlockchainBased Library management and it's characterizati on	Manish verma	Blockchain-Based library is the genesis for the nextgeneration Library management system.	This fits well with the tasks which every librarian has always done, that is, gathering, preserving, and sharing authoritative information	Scalability is an issue
3.	Distributed library management system based on the Blockchain technology	Juan Cabello B.Sc Gerrit Janben B.Sc Alexander Muhle B.Sc	Users to look up for a book and store their transactions and libraries to look up for current holder of books.	Implementing the library-to-user and user-to-user loan	Some blockchain solutions consume too much energy
4.	Application of blockchain technology in libraries and information centers services	Sanjay Kumar Jha	An extensive review of literature on "smart libraries " was carried to ascertain the emerging technologies in the smart library domain.	Implementating the blockchain features, i.e sending tokens, reading from metadata	Blockchain are some times inefficient

5.	Relevance of blockchain technology and the management of libraries and archives in the 4IR	Adeyinka Tella, Halimah Odunayo Amuda, Yusuf Ayodeji Ajani	Ten system librarians selected from five different libraries and ten archivists selected from five different archives in the southwest geopolitical zone of Nigeria represent the sample for the study.	Creating distinct library and user clints	Not completely secure

3.IDEATION AND PROPOSED SOLUTION

Says

What have we heard them say?
What can we imagine them saying?

concerns about the security and integrity of library records .

Evaluation the technical aspects of block chain integration.

Wondering about the benefits of block chain for library services.

Thinks

What are their wants, needs, hopes, and dreams?
What other thoughts might influence their behavior?

excitement about the potential for improved record keeping and transparency

hopeful for improved access to library resources

concerned about the complexity and potential technical issues

Blockchain Powered Library Management

Does

What behavior have we observed?
What can we imagine them doing?

learning how to use blockchain technology for library management

exploring the new features enabled by blockchain

setting up and maintaining blockchain infrastructure

Feels

What are their fears, frustrations, and anxieties?
What other feelings might influence their behavior?

training and support for blockchain implementation

user -friendly interfaces technical training and support

cost - benefit analysis

See an example

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- **10 minutes** to prepare
- **1 hour** to collaborate
- **2-8 people** recommended

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

How might we [your problem statement]?

Key rules of brainstorming

To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Person 1

- use blockchain to create a decentralized catalog of library resources
- Blockchain ensures that once a book is added the library's collection ,the record is immutable,reducing the risk of data manipulation
- Integrate blockchain to manage digital rights for e-books,ensuring that only authorized users can access and borrow digital materials

person 2

- Store digital assets in a decentralized manner using blockchain
- Implement smart contracts to facilitate the borrowing and return of physical and digital books
- Create a token - based reward system for active library users

person 3

- Blockchain can streamline interlibrary loans by securely recording and sharing information between library
- Use blockchain to ensure the long - term preservation of digital resources
- Blockchain can enhance the privacy of user data and transaction

person 4

- The transparent nature of blockchain can build trust between the library and ratings to help users discovers new content
- Implement a blockchain - based system for crowdsourced book review and
- Use blockchain to track and manage intellectual property rights for authors and publishers

integrate blockchain to manage digital rights for e books,ensureing that only authorized users can access and borrow digital materials

Create a token - based reward system for active library users

Use blockchain to track and manage intellectual property rights for authors and publishers

The transparent nature of blockchain can build trust between the library and ratings to help users discovers new content

Blockchain can streamline interlibrary loans by securely recording and sharing information between library

Implement smart contracts to facilitate the borrowing and return of physical and digital books

Blockchain ensures that once a book is added the library's collection ,the record is immutable,reducing the risk of data manipulation

Use blockchain to ensure the long - term preservation of digital resources

4

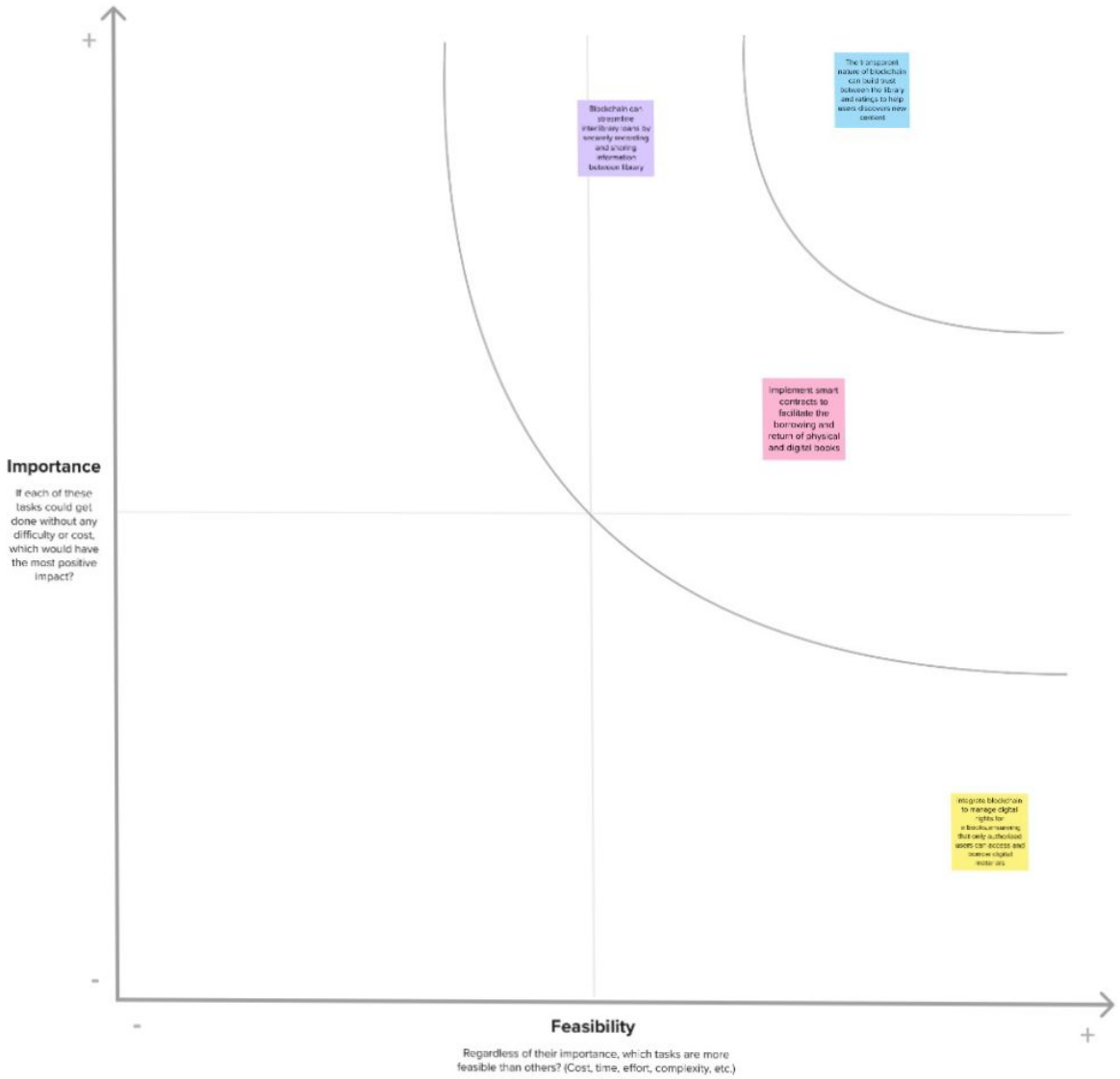
Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.



After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

A **Share the mural**
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

B **Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

Share template feedback

4.REQUIREMENT ANALYSIS

Date	24 oct 2023
Team ID	NM2023TMlD03178
Project name	Blockchain Powered LibraryManagement

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR NO.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-I	user Registration	Users should be able to register and create accounts
FR.2	user Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Search and discovery	Implement a search feature for users to find books.
FR-4	Notification system	Notify users about due dates, reservations, and im r tant libra u dates.
FR.s	Interlibrary Loan	Enable users to request books from other libraries
FR-6	Smart Contracts	Implement smart contracts ror book lending, return, nd a ment r cess

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR NO.	Non-Fu nct ional Requirement	Description
NFR-I	usability	Providing an intuitive and user-friendly interface for both [brary staff and patrons.
NFR-2	Security	Ensuring the integrity and confidentiality of library records, user data. and transaction history through encryption
NFR-3	Reliability	The Should be available and responsive. minimizi downtime or service disru lions.
NFR.4	Performance	Ensuring tast transaction processing and response times, especially during high demand periods,
NFR-S	Availability	Maintaining a transparent and immutable record or library transactions for auditing purposes.
NFR-6	Scalability	The system should be able to handle a growing number of users, books, and transactions

Project Design Phase-II
Data Flow Diagram & User Stories

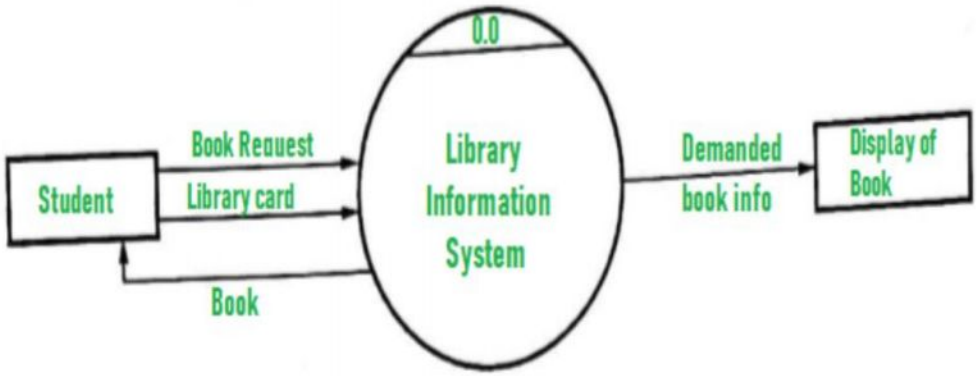
5.PROJECT DESIGN

Date	24-10-23
Team Id	NM2023TMID03178
Project Name	Block chain – Powered Library Management

Data Flow Diagrams:

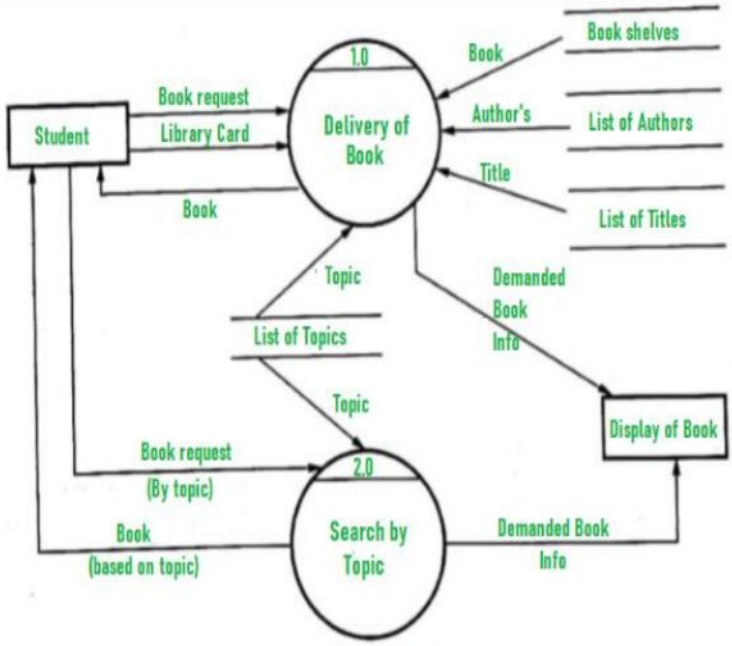
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored

1. Level 0 DFD –



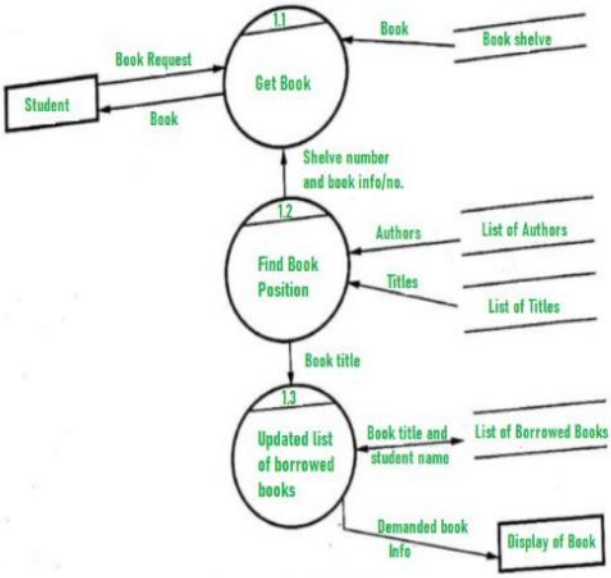
Project Design Phase-II
Data Flow Diagram & User Stories

2.Level 1 DFD-



Level 1 DFD

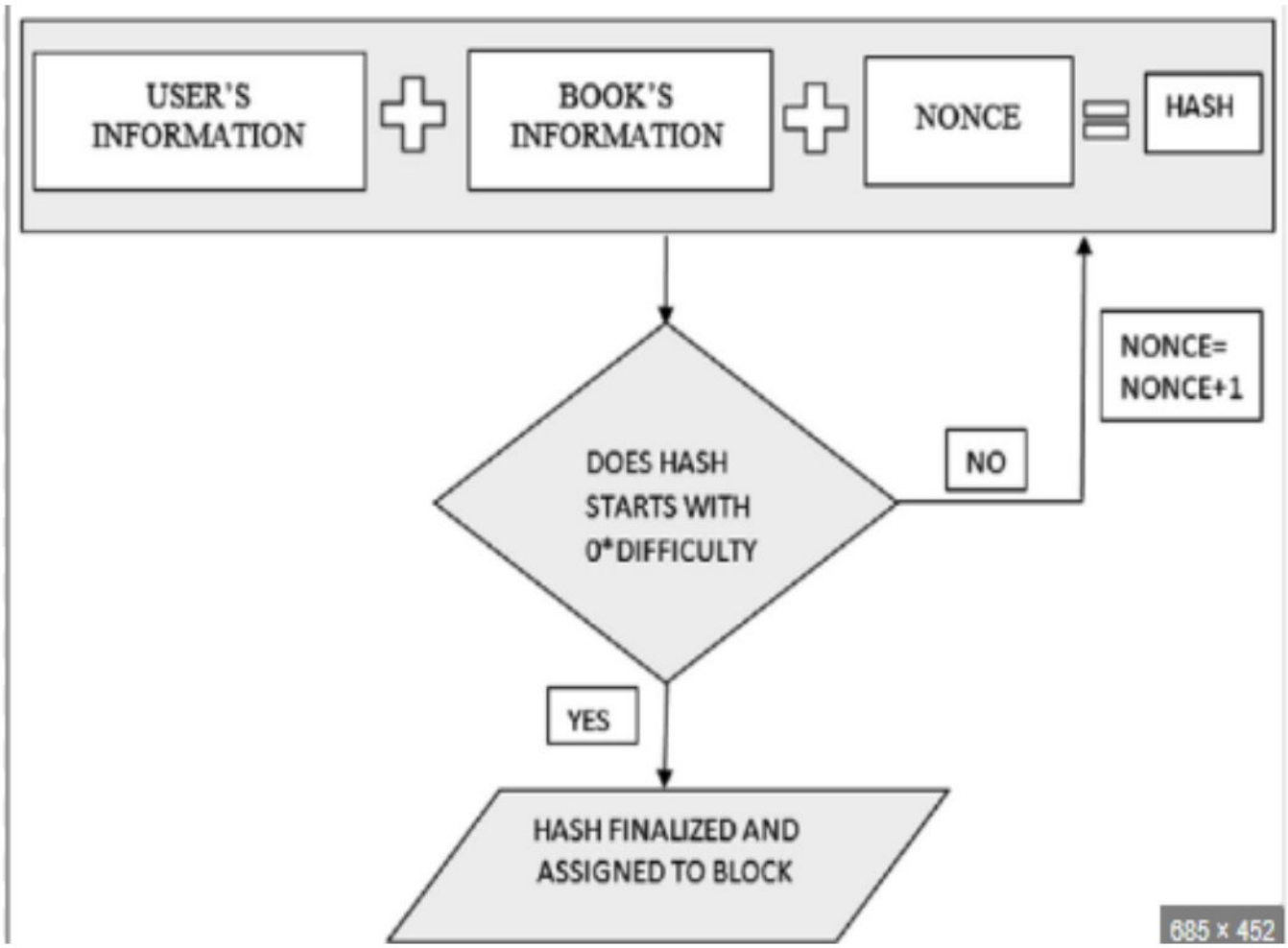
3. Level 2 DFD –



Level 2 DFD

Project Design Phase-II
Data Flow Diagram & User Stories

User services:



Date	24-10-2023
Team ID	NM2023TMID03178
Project name	Block chain-Powered Library Management

SOLUTION ARCHITECTURE

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed,

Project Design Phase – I
Solution Architecture

```
+-----------------------------------+
          |   User Interfaces (Web/Mobile)  |
          +----------------+------------------+
                    |
            +-----------v-----------+
            |   Frontend Components  |
            +-----------------------+
                    |
    +--------------------------v------------------+
    |                            |
+-----v-----+         +----------------+   +---v----+
|  Users  |          |  Library Staff |    | Blockchain|
+---------+         +----------------+   |  Layer   |
    |                |         +----------+
    |                |              |
+-----v-----+         +---v----+           |
| User Reg |          | Catalog|           |
| & Auth   |          | Search |           |
+---------+         +---+----+           |
    |              |               |
    |              |               |
+-----v-----+      +---v--+           |
| Database|       |  Library|           |
| Layer   |       |  System |           |
+---------+      +---+----+           |
                |              |
+------------------------v-------------------+
```

Project Design Phase – I
Solution Architecture

```
| Authentication and Authorization          |
|              |                  |
|   +--------------v--------------+        |
|   | Key Features          |        |
|   |                   |        |
|   +-------------------------+        |
|                  |           |
+------------------------v---------------+
| Security            |
|                  |
|          +-----------------+
|          | Scalability   |
|          +-----------------+
|                  |
|   +--------------------v-----------------+
|   | External Service Integration        |
|   |               |               |
|   +--------------------v-----------------+
|                  |
|   +--------------------v-----------------+
|   | Analytics and Reporting             |
|   |               |               |
|   +--------------------v-----------------+
|                  |
|   +--------------------v-----------------+
|   | Maintenance and Upkeep              |
|   |               |               |
|   +--------------------v-----------------+
```

```
|                    |
|   +--------------------v----------------+
|   | User Training and Support           |
|   |                    |                |
|   +--------------------v----------------+
|                    |
|   +--------------------v----------------+
|   | Compliance and Legal Considerations |
|   |                    |                |
|   +--------------------v----------------+
|                    |
|   +--------------------v----------------+
|   | Monitoring and Logging              |
|   |                    |                |
|   +--------------------v----------------+
|                    |
|   +--------------------v----------------+
|   | Testing                             |
|   |                    |                |
|   +--------------------v----------------+
|                    |
|   +--------------------v----------------+
|   | Deployment and Scaling              |
|   |                    |                |
|   +--------------------v----------------+
|                    |
|   +--------------------v----------------+
|   | Documentation                       |
```

6.PROJECT PLANNING &SCHEDULING

TECHNICAL ARCHITECTURE

1. **Blockchain Technology**: Choose a blockchain platform like Ethereum, Hyperledger Fabric, or a suitable alternative that fits your library's needs. Ethereum, for example, is often used for decentralized applications.

2. **Smart Contracts**: Develop smart contracts to manage library transactions, such as borrowing and returning books, user registration, and book cataloging. Smart contracts automate these processes and are executed on the blockchain.

3. **Decentralized Identity**: Implement decentralized identity solutions to manage user authentication and authorization. This ensures data privacy and security while granting users control over their personal information.

4. **Tokenization**: Create a library-specific token or use an existing cryptocurrency to facilitate payments, fines, and rewards within the library ecosystem.

5 **Data Storage**: Store book catalog information, user data, and transaction history on the blockchain or in a decentralized storage system. IPFS (InterPlanetary File System) is a popular choice for decentralized storage.

SPRINT PLANNING & ESTIMATION

Sprint 1: Project Initiation

Tasks:

- Project kickoff meeting.
- Identify project stakeholders.
- Define the vision and objectives.
- Assemble the project team.

- Set up project management tools.

- High-level project plan and roadmap.

Sprint 2: Requirement Gathering and Analysis

* **Tasks**:

- Detailed requirements gathering from library staff and users.

- User stories, use cases, and workflow documentation.

- Data modeling for book catalog, user profiles, and transactions.

- Privacy and security requirements.

- Requirement prioritization and backlog creation.

Sprint 3: Architecture and Technology Selection

* **Tasks**:

- Select the blockchain platform (e.g., Ethereum, Hyperledger).

- Define the system architecture.

- Develop the smart contract design.

- Decide on the decentralized identity solution.

- Identify external systems for integration.

Sprint 4: Smart Contract Development

* **Tasks**:

- Develop smart contracts for book cataloging, borrowing, and returning.

- Create a library-specific token or use an existing cryptocurrency.

- Ensure smart contracts are secure and efficient.

Sprint 5: User Interface Development

* **Tasks**:

- Design and develop user interfaces for patrons and library staff.

- Implement user registration and authentication.

- Integrate book search and borrowing functionality.

Sprint 6: Decentralized Identity and Data Storage

* **Tasks**:

- Implement decentralized identity for user management.

- Set up decentralized data storage (e.g., IPFS) for book information and user profiles.

Sprint 7: Integration and Testing

* **Tasks**:

- Integrate the blockchain library system with external libraries and systems.

- Conduct thorough testing, including unit, integration, and security testing.

- Address any issues and bugs.

Sprint 8: Privacy and Security Enhancements

* **Tasks**:

- Enhance privacy features using techniques like zero-knowledge proofs.

- Strengthen security measures to protect user data.

- Conduct security audits and penetration testing.

Sprint 9: Scalability and Performance Optimization

* **Tasks**:

- Implement strategies for scalability, like sidechains or layer-2 solutions.

- Optimize the system's performance.

- Conduct load testing to ensure it can handle increased user loads.

Sprint 10: Documentation, Training, and Deployment

* **Tasks**:

- Create comprehensive documentation for the system, including user guides and developer documentation.

- Provide training to library staff and users.

- Deploy the blockchain library management system in a controlled environment.

SPRINT DELIVERY SCHEDULE

Sprint 1: Project Initiation and Planning

* **Deliverables:

- Project kickoff meeting and charter.

- Stakeholder identification.

- High-level project plan and roadmap.

Sprint 2: Requirement Gathering and Analysis

Deliverables**:

- Detailed requirements documentation.

- User stories, use cases, and workflow diagrams.

- Prioritized backlog.

Sprint 3: Architecture and Technology Selection

* **Deliverables**:

- Selection of the blockchain platform.

- Initial system architecture design.

- Smart contract design outline.

Sprint 4: Smart Contract Development (Part 1)

Deliverables:

- Initial smart contracts for book cataloging.

- Basic user registration and authentication.

Sprint 5: Smart Contract Development (Part 2)

Deliverables:

- Complete smart contracts for book borrowing and returning.

- Library-specific token integration.

Sprint 6: User Interface Development

Deliverables:

- User interfaces for patrons and library staff.

- Functional user registration and authentication.

Sprint 7: Decentralized Identity and Data Storage

Deliverables:

- Implementation of decentralized identity solutions.

- Integration of decentralized data storage for book information.

Sprint 8: Integration and Testing

Deliverables:

- Integration with external library systems.

- Comprehensive testing, including security testing.

Sprint 9: Privacy and Security Enhancements

Deliverables:

- Enhanced privacy features and security measures.

- Security audits and penetration test results.

Sprint 10: Scalability and Performance Optimization

Deliverables:

- Implemented scalability strategies.

- Optimized system performance.

Sprint 11: Documentation and Training

Deliverables:

- Comprehensive system documentation.

- User training materials.

Sprint 12: Deployment and Final Testing

Deliverables:

- Deployment of the blockchain library management system in a controlled environment.

- Final system testing and user acceptance testing.

7.CODING AND SOLUTIONING

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

contract LibraryManagement {
  address public owner;

  // Book structure
  struct Book {
    string title;
    string author;
    bool available;
  }

  // Mapping to store books
  mapping(uint256 => Book) public books;
  uint256 public bookCount;

  // Events for tracking actions
  event BookAdded(uint256 bookId, string title, string author);
  event BookBorrowed(uint256 bookId, address borrower);
  event BookReturned(uint256 bookId, address borrower);

  // Constructor to set the contract owner
  constructor() {
    owner = msg.sender;
  }
```

```
// Modifier to restrict access to the owner
modifier onlyOwner() {
    require(msg.sender == owner, "Only the contract owner can perform this action");
    _;
}

// Add a new book to the library
function addBook(string memory title, string memory author) public onlyOwner {
    uint256 bookId = bookCount;
    books[bookId] = Book(title, author, true);
    bookCount++;
    emit BookAdded(bookId, title, author);
}

// Borrow a book
function borrowBook(uint256 bookId) public {
    require(bookId < bookCount, "Invalid book ID");
    require(books[bookId].available, "Book is not available");

    books[bookId].available = false;
    emit BookBorrowed(bookId, msg.sender);
}

// Return a borrowed book
function returnBook(uint256 bookId) public {
    require(bookId < bookCount, "Invalid book ID");
    require(!books[bookId].available, "Book is already available");
```

```
        books[bookId].available = true;
        emit BookReturned(bookId, msg.sender);
    }
}
```

SOLUTIONING

Home | Library Management.sol

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

contract BookRegistry {
    address public owner;

    constructor() {    997874 gas 972600 gas
        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;

```

0 listen on all transactions Search with transaction hash or address

[vm] from: 0x5B3...eddC4 to: BookRegistry.(constructor) value: 0 wei data: 0x608...20033 logs: 0 hash: 0x835...4d525 Debug

Confirm transaction

You are about to create a transaction on Main Network. Confirm the details to send the info to your provider.
The provider for many users is MetaMask. The provider will ask you to sign the transaction before it is sent to Main Network.

From: 0x95d1D247465fC4d91585f6e8d1870ae1B8544826
To: (Contract Creation)
Data: 0x6080604052348015610010576000080fd5b50336000080610...
Amount: 0 Ether

Gas estimation: 0x2dc6c0
Gas limit: 0x2dc6c0
Max Priority fee: 1 Gwei
Max fee (Not less than base fee 62.414813396 Gwei): 62.414813 Gwei
Max transaction fee: 0.187244440188 Ether

Do not show this warning again.

Confirm Cancel

Windows PowerShell

```
Microsoft Windows [Version 10.0.19045.3324]
(c) Microsoft Corporation. All rights reserved.

D:\8_Problem_Statement_8_Book_details (1)\Book-details\book-details\src>npm install
npm WARN deprecated stable@0.1.8: Modern JS already guarantees Array#sort() is a stable sort, so this library is deprecated. See the compatibility table on MDN: https:/
/developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/sort#browser_compatibility
npm WARN deprecated rollup-plugin-terser@7.0.2: This package has been deprecated and is no longer maintained. Please use @rollup/plugin-terser
npm WARN deprecated sourcemap-codec@1.4.8: Please use @jridgewell/sourcemap-codec instead
npm WARN deprecated w3c-hr-time@1.0.2: Use your platform's native performance.now() and performance.timeOrigin.
npm WARN deprecated workbox-cacheable-response@6.6.0: workbox-background-sync@6.6.0
npm WARN deprecated svgo@1.3.2: This SVGO version is no longer supported. Upgrade to v2.x.x.

added 1567 packages, and audited 1568 packages in 8m

277 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (2 moderate, 6 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

D:\8_Problem_Statement_8_Book_details (1)\Book-details\book-details\src>
D:\8_Problem_Statement_8_Book_details (1)\Book-details\book-details\src>npm install bootstrap

added 1 package, and audited 1569 packages in 49s

278 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (2 moderate, 6 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
```

Type here to search 33°C ENG 21:29

swarnashree7603/FOOD-TRACI

Remix - Ethereum IDE

remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.18+commit.87f61d96.js

Sabarimala Online (... Gmail YouTube Maps

DEPLOY & RUN TRANSACTIONS

ENVIRONMENT

Injected Provider - MetaMask

Main (1) network

ACCOUNT

0x95d...44826 (0 ether)

GAS LIMIT

3000000

VALUE

0 Wei

CONTRACT

BookRegistry - Library Management.s

evm version: paris

Deploy

Publish to IPFS

At Address Load contract from Address

Transactions recorded 1

Deployed Contracts

Home Library Management.sol

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

contract BookRegistry {
    address public owner;

    constructor() {    997874 gas 972680 gas
        owner = msg.sender;
    }

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

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

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

```

0 listen on all transactions Search with transaction hash or address

creation of BookRegistry pending...

MetaMask Notification

Ethereum Mainnet

Account 1 New contract

https://remix.ethereum.org

CONTRACT DEPLOYMENT

$0.00

DETAILS DATA

Network is busy. Gas prices are high and estimates are less accurate.

Site suggested

Gas (estimated) $339.21 0.18724444 ETH

Very likely in < 15 seconds Max fee: 0.18724444 ETH

Total $339.21 0.18724444 ETH

Amount + gas fee Max amount: 0.18724444 ETH

Type here to search

33°C Partly sunny ENG 21:06 25-10-2023

```
D:\8_Problem_Statement_8_Book_details (1)\Book-details\book-details\src>
D:\8_Problem_Statement_8_Book_details (1)\Book-details\book-details\src>npm install bootstrap

added 1 package, and audited 1569 packages in 49s

278 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (2 moderate, 6 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

D:\8_Problem_Statement_8_Book_details (1)\Book-details\book-details\src>
D:\8_Problem_Statement_8_Book_details (1)\Book-details\book-details\src>npm start

> book-details@0.1.0 start
> react-scripts start

(node:2052) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' opti
on.
(Use `node --trace-deprecation ...` to show where the warning was created)
(node:2052) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' op
tion.
Starting the development server...

One of your dependencies, babel-preset-react-app, is importing the
"@babel/plugin-proposal-private-property-in-object" package without
declaring it in its dependencies. This is currently working because
"@babel/plugin-proposal-private-property-in-object" is already in your
node_modules folder for unrelated reasons, but it may break at any time.

babel-preset-react-app is part of the create-react-app project, which
is not maintianed anymore. It is thus unlikely that this bug will
ever be fixed. Add "@babel/plugin-proposal-private-property-in-object" to
your devDependencies to work around this error. This will make this message
go away.
```

Type here to search
33°C ENG 21:29 25-10-2023

OLIDITY COMPILER

COMPILER

0.8.18+commit.87f61d96

Include nightly builds

Auto compile

Hide warnings

Advanced Configurations

Compile Library Managemen...

Compile and Run script

CONTRACT

BookRegistry (Library Management.sc

Publish on Ipfs

Publish on Swarm

Compilation Details

ABI Bytecode

Home

Library Management.sol

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

contract BookRegistry {
    address public owner;

    constructor() {    997874 gas 972600 gas
        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;

```

0 listen on all transactions

Search with transaction hash or address

creation of BookRegistry pending...

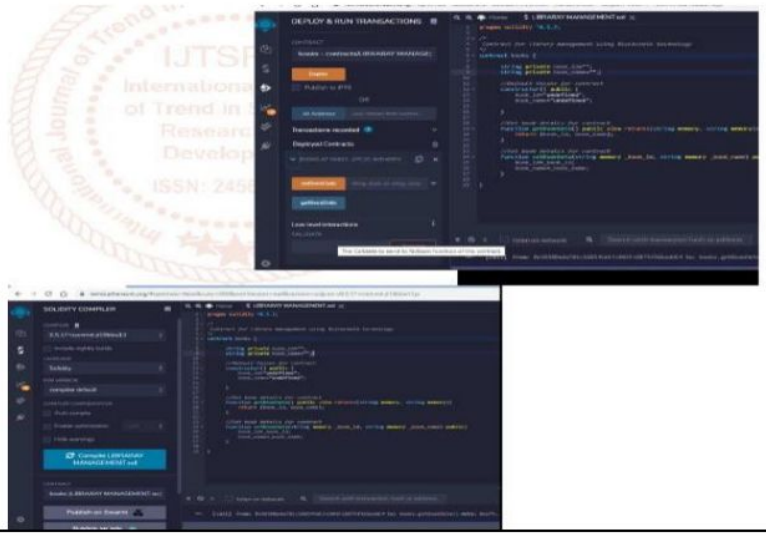
8.PERFORMANCE TESTING

Performance Test

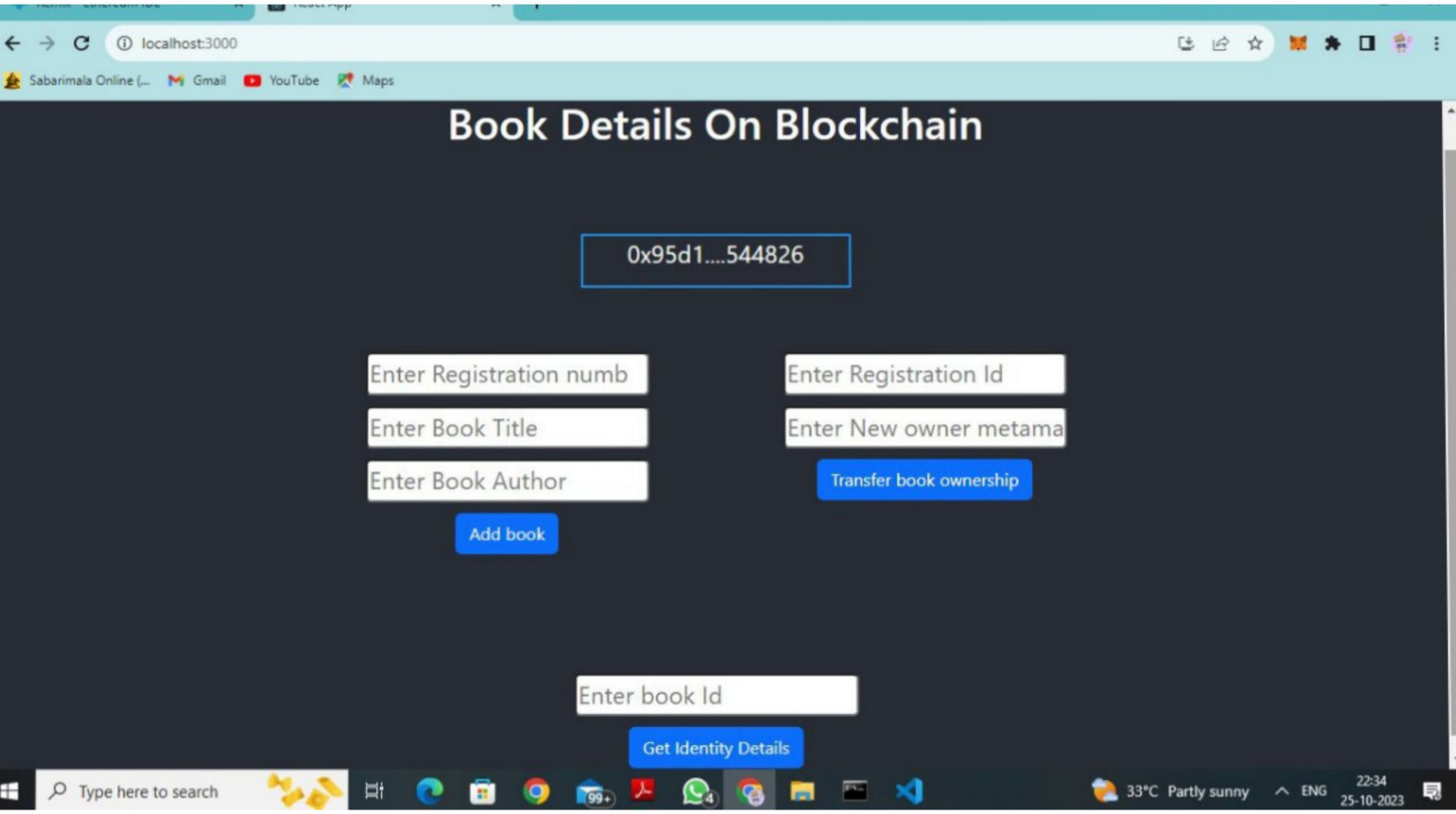
Date	24 Oct 2023
Team ID	NM2023TMID03178
Project Name	Blockchain Powered LibraryManagement

Model Performance Testing:

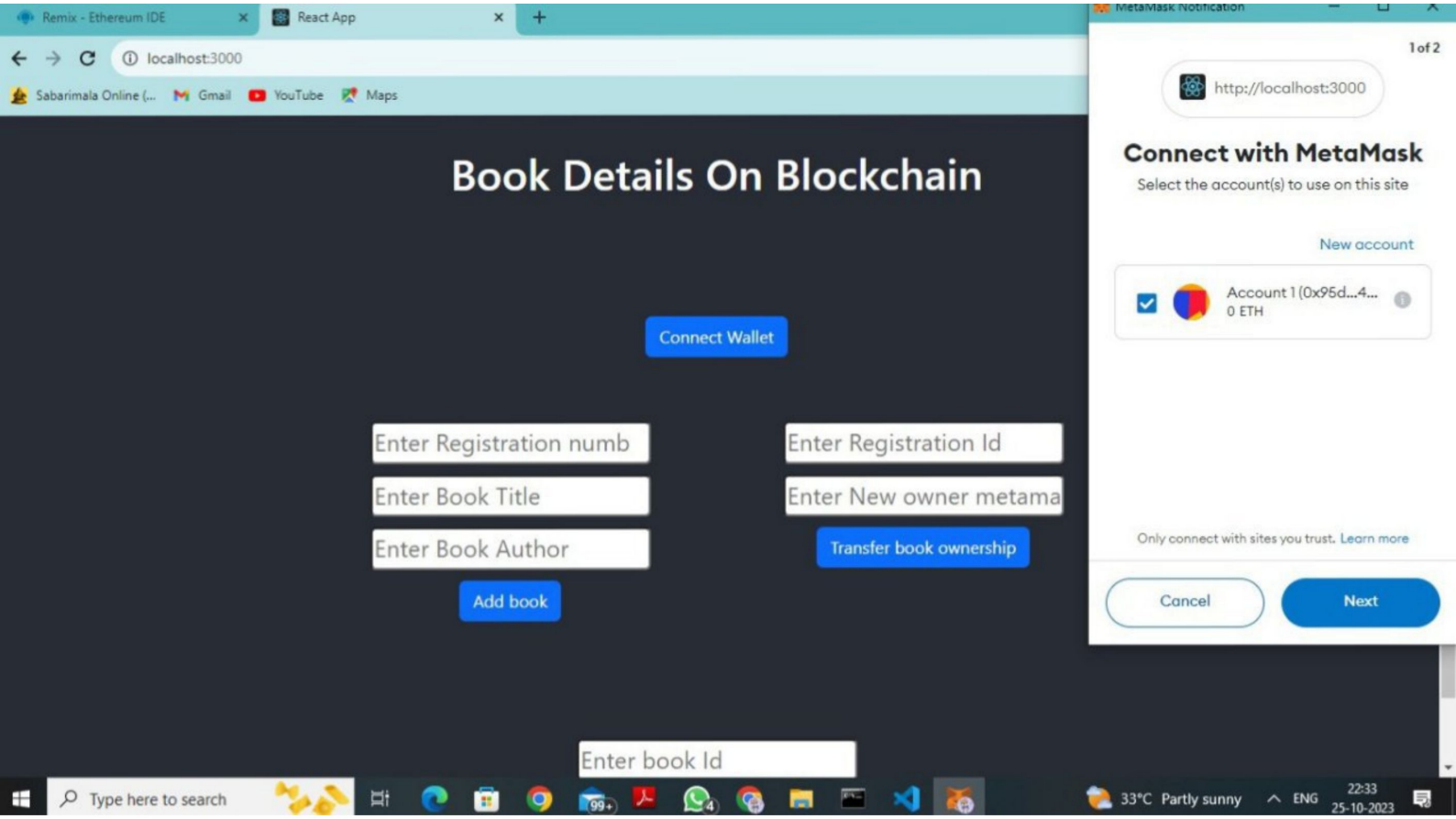
Project team shall fill the following information in the performance testing template.

Parameter	Values	Screenshot
Metrics	Library management using Blockchain technology	

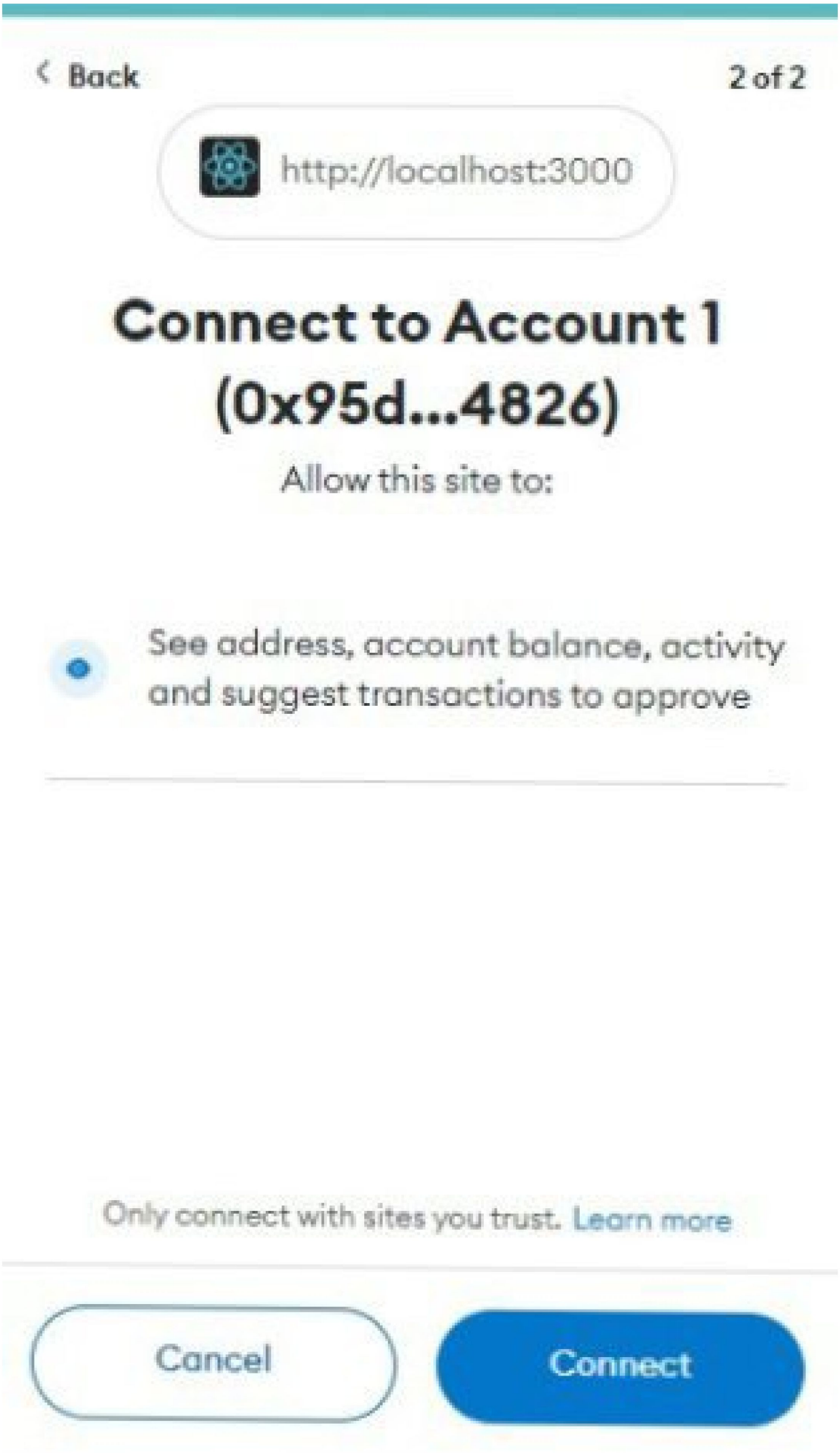
9.RESULTS



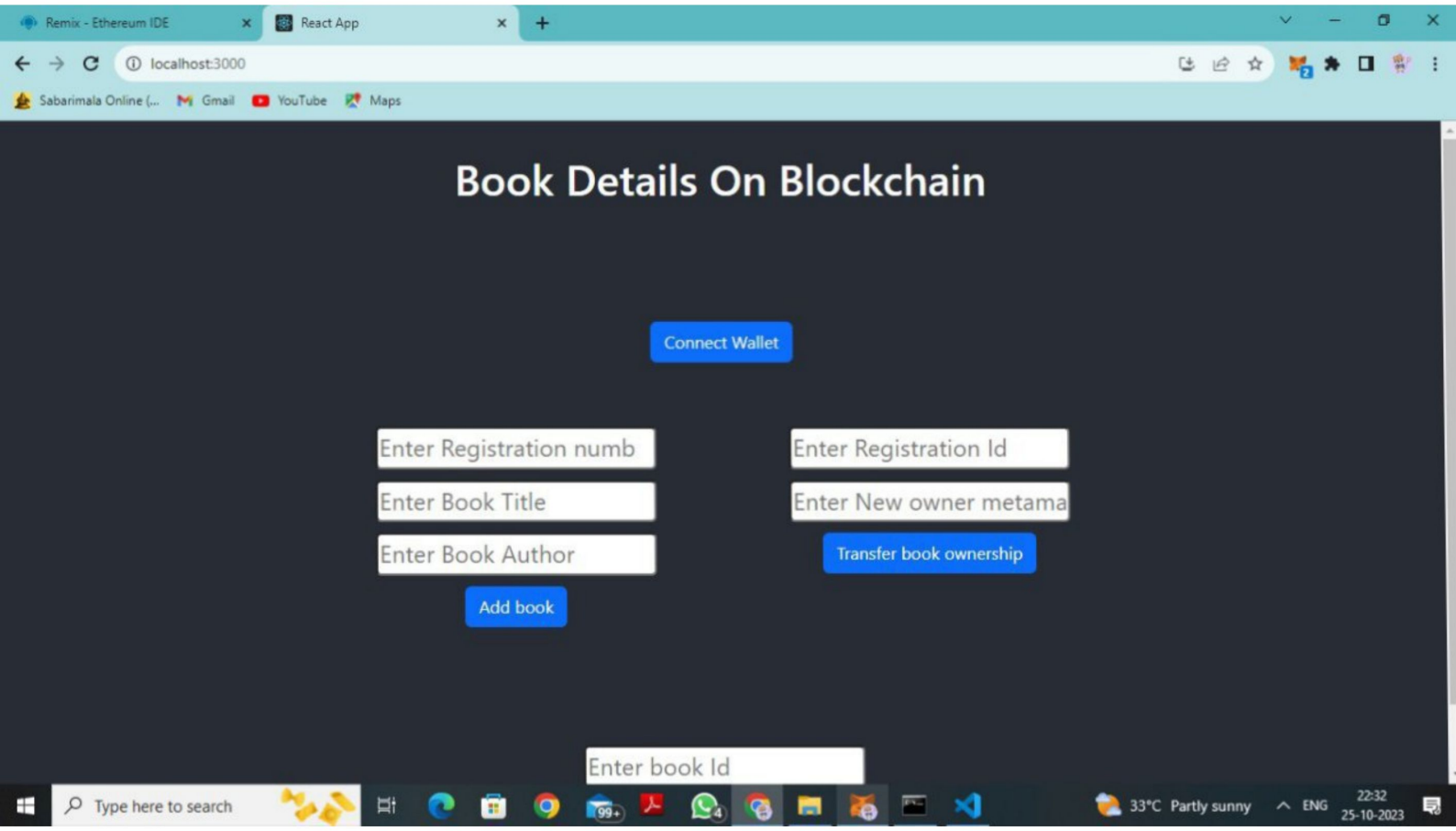
Scanned by TapScanner



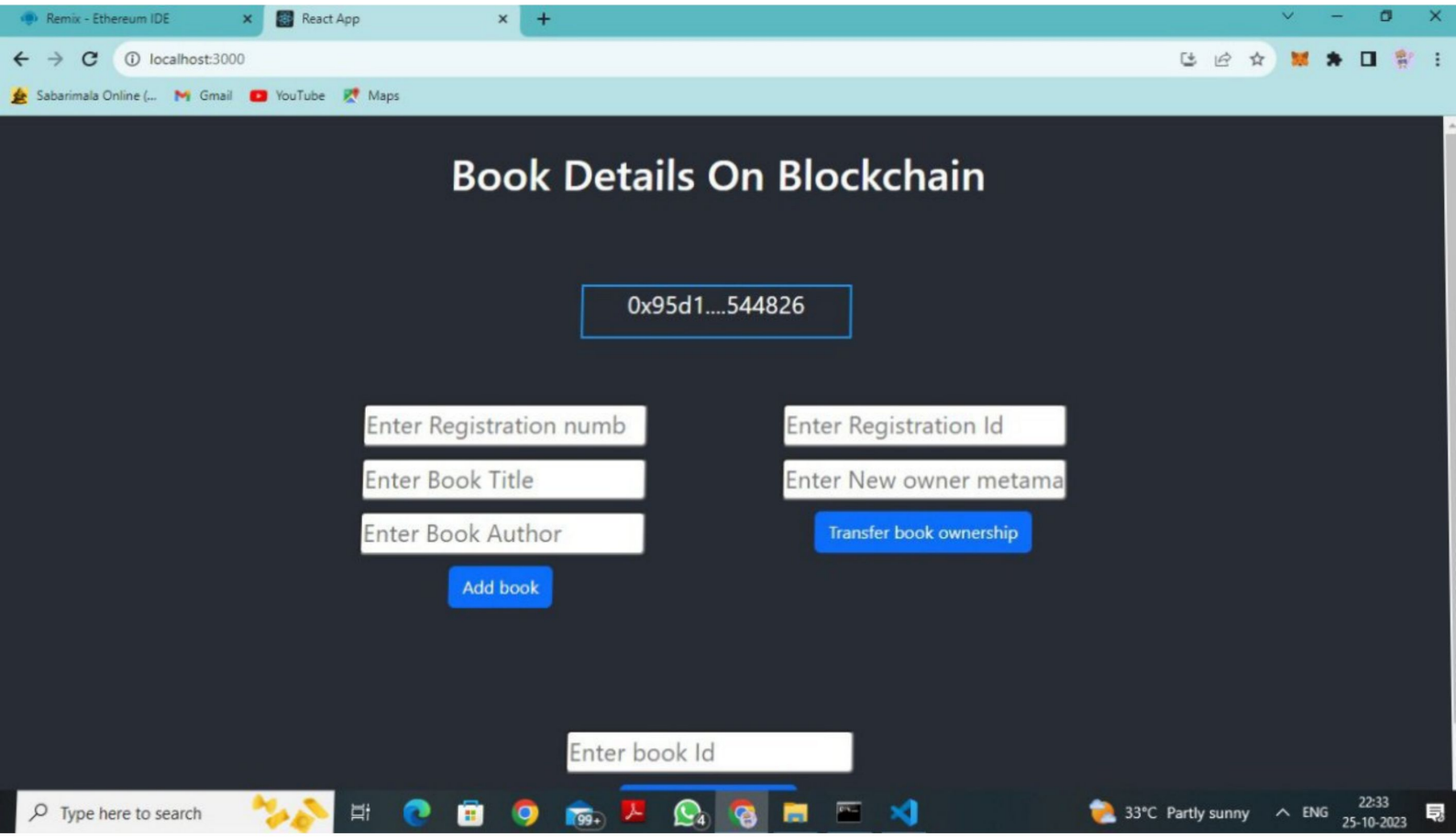
Scanned by TapScanner



Scanned by TapScanner



Scanned by TapScanner



Scanned by TapScanner

10.ADVANTAGES

- **Enhanced Security: Blockchain technology provides a high level of security through cryptography and decentralization, making it extremely difficult for unauthorized access and data tampering.**
- **Data Integrity: Records stored on a blockchain are immutable, ensuring the integrity of library data, which is crucial for maintaining accurate resource records.**
- **User Privacy: Blockchain allows for greater user privacy control, as users can manage their own data, reducing the risk of privacy breaches.**
- **Streamlined Operations: Smart contracts can automate processes such as book lending and returns, reducing administrative overhead and improving operational efficiency**
- **Reduced Costs: Automation, reduced administrative tasks, and enhanced security can lead to cost savings in library operations.**

11.DISADVANTAGES

- **Complex Implementation: Integrating blockchain technology can be complex and requires technical expertise. Libraries may face challenges in terms of initial setup and maintenance.**
- **Scalability: As the blockchain network grows, scalability becomes an issue. Ensuring that the system can handle a large number of transactions and records can be a challenge.**

- **Cost of Entry: Implementing a blockchain system can be expensive in terms of infrastructure and development costs, which may be a barrier for smaller libraries.**
- **User Learning Curve: Both library staff and users may need to adapt to the new system, which can lead to a learning curve and resistance to change.**
- **Regulatory and Legal Challenges: Compliance with existing regulations, especially concerning user data and privacy, can be complex in a blockchain-powered system.**

12.CONCLUSION

In conclusion, Blockchain Powered Library Management represents a promising and innovative approach to modernizing traditional libraries in the digital age. This technology offers a multitude of advantages, including enhanced security, data integrity, user privacy, streamlined operations, cross-library resource sharing, transparency, and cost savings. By leveraging blockchain, libraries can adapt to the changing needs of their patrons while preserving their core mission of knowledge dissemination.

13.FUTURE SCOPE

Interconnected Libraries

Decentralized Cataloging

Tokenization for Reward

Integration with E-books and Digital Media

Immutable Digital Archives

Advanced Analytics

NFTs for Rare Collections

Blockchain-Based Digital Preservation

13.APPENDIX

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

contract LibraryManagement {
    address public owner;

    // Book structure
    struct Book {
        string title;
        string author;
        bool available;
    }

    // Mapping to store books
    mapping(uint256 => Book) public books;
    uint256 public bookCount;

    // Events for tracking actions
    event BookAdded(uint256 bookId, string title, string author);
    event BookBorrowed(uint256 bookId, address borrower);
    event BookReturned(uint256 bookId, address borrower);

    // Constructor to set the contract owner
    constructor() {
        owner = msg.sender;
    }
```

```
// Modifier to restrict access to the owner
modifier onlyOwner() {
    require(msg.sender == owner, "Only the contract owner can perform this action");
    _;
}

// Add a new book to the library
function addBook(string memory title, string memory author) public onlyOwner {
    uint256 bookId = bookCount;
    books[bookId] = Book(title, author, true);
    bookCount++;
    emit BookAdded(bookId, title, author);
}

// Borrow a book
function borrowBook(uint256 bookId) public {
    require(bookId < bookCount, "Invalid book ID");
    require(books[bookId].available, "Book is not available");

    books[bookId].available = false;
    emit BookBorrowed(bookId, msg.sender);
}

// Return a borrowed book
function returnBook(uint256 bookId) public {
    require(bookId < bookCount, "Invalid book ID");
```

```
        require(!books[bookId].available, "Book is already available");

        books[bookId].available = true;
        emit BookReturned(bookId, msg.sender);
    }
}
```

SOURCE CODE


```
[
    {
        "inputs": [],
        "stateMutability": "nonpayable",
        "type": "constructor"
    },
    {
        "anonymous": false,
        "inputs": [
            {
                "indexed": false,
                "internalType": "uint256",
                "name": "bookId",
                "type": "uint256"
```

```
                },
                {
                        "indexed": false,
                        "internalType": "string",
                        "name": "title",
                        "type": "string"
                },
                {
                        "indexed": false,
                        "internalType": "string",
                        "name": "author",
                        "type": "string"
                }
        ],
        "name": "BookAdded",
        "type": "event"
},
{
        "anonymous": false,
        "inputs": [
                {
                        "indexed": false,
                        "internalType": "uint256",
                        "name": "bookId",
```

```
                "type": "uint256"
            },
            {
                "indexed": false,
                "internalType": "address",
                "name": "borrower",
                "type": "address"
            }
        ],
        "name": "BookBorrowed",
        "type": "event"
    },
    {
        "anonymous": false,
        "inputs": [
            {
                "indexed": false,
                "internalType": "uint256",
                "name": "bookId",
                "type": "uint256"
            },
            {
                "indexed": false,
                "internalType": "address",
```

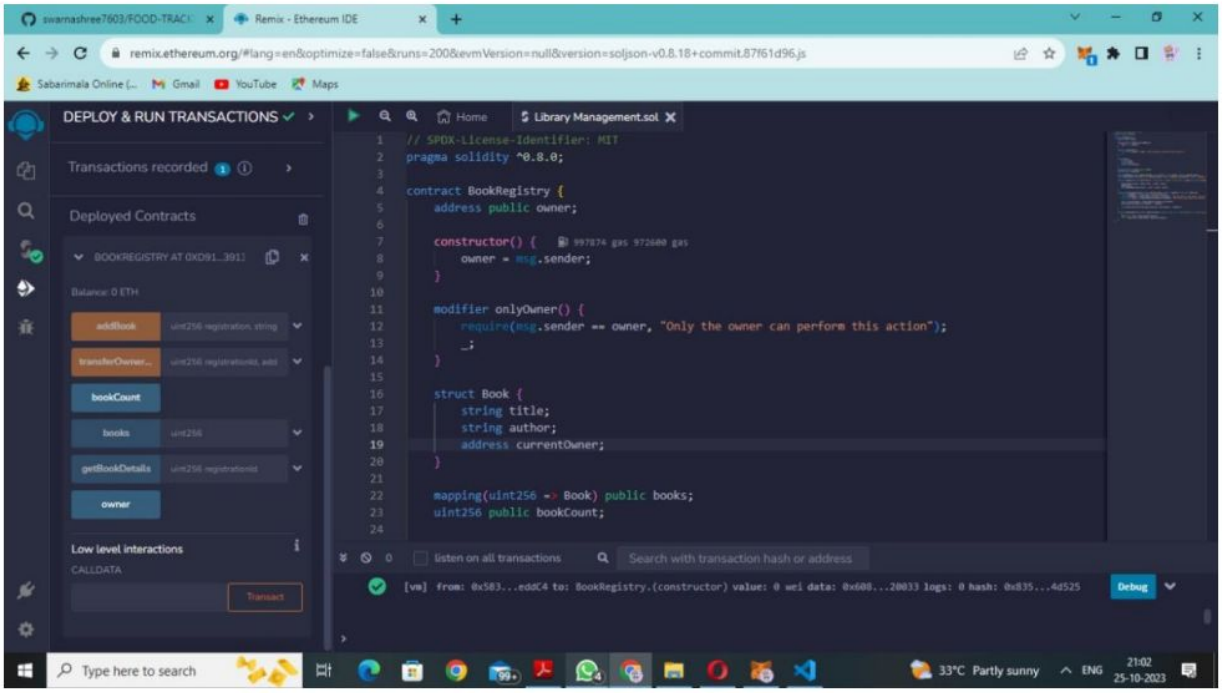
```
                "name": "borrower",
                "type": "address"
            }
        ],
        "name": "BookReturned",
        "type": "event"
    },
    {
        "inputs": [
            {
                "internalType": "string",
                "name": "title",
                "type": "string"
            },
            {
                "internalType": "string",
                "name": "author",
                "type": "string"
            }
        ],
        "name": "addBook",
        "outputs": [],
        "stateMutability": "nonpayable",
        "type": "function"
```

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

```
            {
                "internalType": "string",
                "name": "title",
                "type": "string"
            },
            {
                "internalType": "string",
                "name": "author",
                "type": "string"
            },
            {
                "internalType": "bool",
                "name": "available",
                "type": "bool"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [
            {
                "internalType": "uint256",
                "name": "bookId",
```

```
                "type": "uint256"
            }
        ],
        "name": "borrowBook",
        "outputs": [],
        "stateMutability": "nonpayable",
        "type": "function"
    },
    {
        "inputs": [],
        "name": "owner",
        "outputs": [
            {
                "internalType": "address",
                "name": "",
                "type": "address"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [
            {
```

```
                "internalType": "uint256",
                "name": "bookId",
                "type": "uint256"
            }
        ],
        "name": "returnBook",
        "outputs": [],
        "stateMutability": "nonpayable",
        "type": "function"
    }
]
```

GitHub & Project Demo Link

https://drive.google.com/file/d/1EmhWeNN5fmxefUTJgH0qvjkvvpxq2PAP/view?usp=drivesdk