Transparent Education Data Management

1. INTRODUCTION

1.1 Project Overview

In the ever-evolving landscape of education, the management of student data has become a pressing concern. Traditional systems often fall short in providing a secure and transparent solution for safeguarding educational records. Our project, "Transparent Education Data Management using Blockchain," tackles this challenge head-on by harnessing the power of blockchain technology. This innovative approach aims to create a secure and immutable system for managing educational data, ensuring the utmost integrity and accessibility of records.

The primary objective of our project is to bolster data security by storing educational records on a decentralized blockchain network, effectively eliminating the risks of unauthorized access and data breaches. Leveraging blockchain's immutability, we guarantee that once data is recorded, it cannot be tampered with, providing an unparalleled level of data integrity. Moreover, our user-friendly web-based platform empowers students to have granular control over their educational records, granting or revoking access securely, thereby ensuring data ownership remains with the students. Additionally, the system allows educational institutions, employers, and relevant entities to quickly and efficiently verify the authenticity of student records, reducing the need for time-consuming manual verification processes. By fostering transparency, trust, and efficiency in education data management, our project aims to reshape the way educational records are handled, offering a promising solution for the evolving needs of the education sector.

1.2 Purpose

The purpose of our project, "Transparent Education Data Management using Blockchain," is to revolutionize the way educational data is handled and secured. In an era of increasing concern over data privacy and security, our project seeks to enhance the protection and transparency of student records. By leveraging blockchain technology, we aim to establish a decentralized and tamper-proof system that ensures the utmost data integrity, security, and user empowerment. This solution not only mitigates the risks of unauthorized data breaches but also streamlines the verification of academic credentials, making it more efficient for educational institutions, employers, and students. With this project, we aspire to create a new standard in education data management, where data is securely owned and controlled by the students, while institutions and employers can easily and confidently verify the authenticity of records. The overarching goal is to instill trust, transparency, and efficiency in the educational data management process, ultimately benefiting all stakeholders within the education sector.

2. EXISTING PROBLEM

2.1 Existing problem

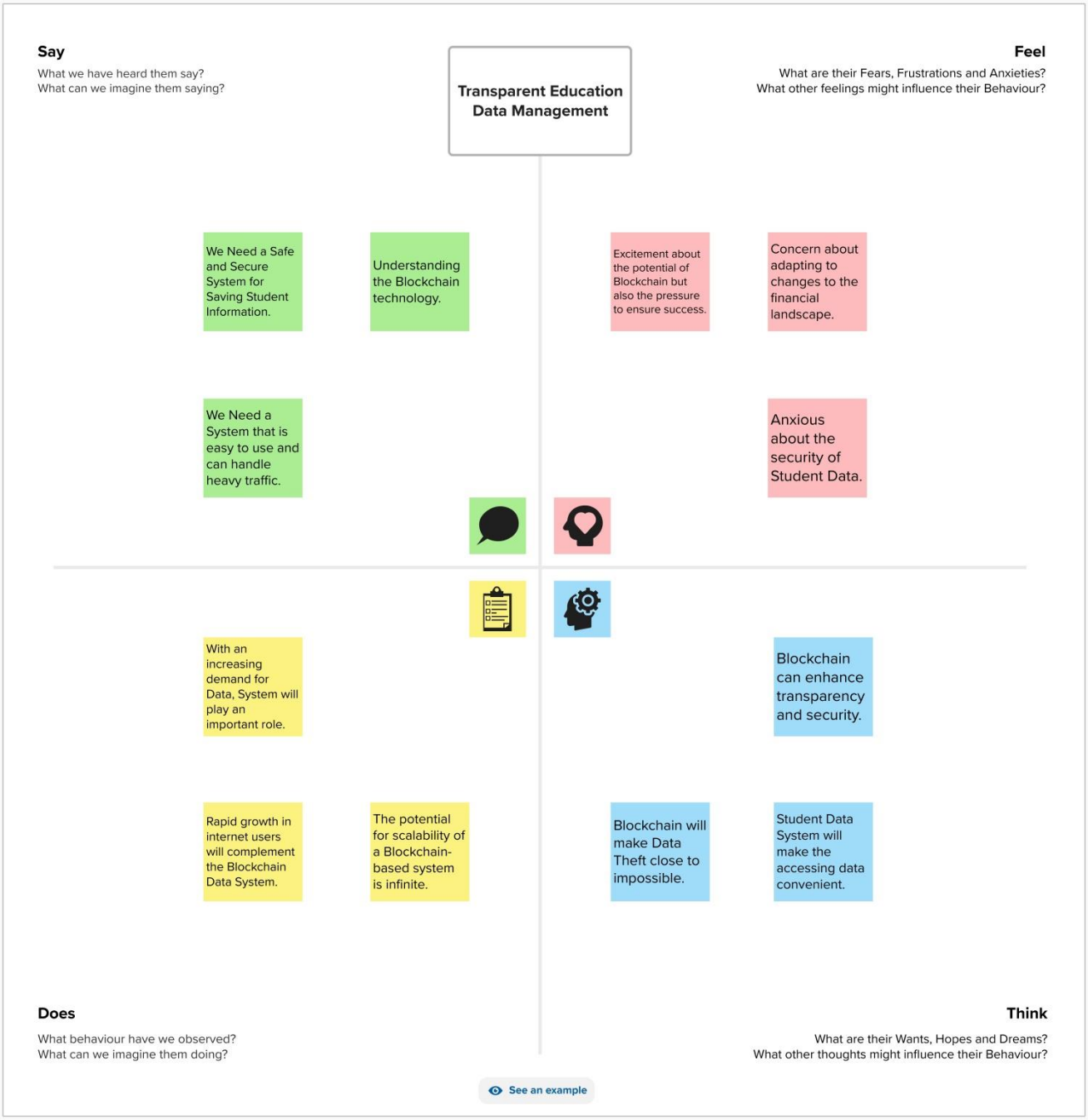
The existing problems in the realm of education data management are multifaceted and pressing. Traditional systems are plagued by security vulnerabilities, making educational records susceptible to unauthorized access, data breaches, and tampering. This compromises the integrity of academic credentials and poses serious privacy concerns for students. Verification processes, often manual and time-consuming, are prone to errors, delays, and inconsistencies, affecting both institutions and students seeking to access opportunities. Additionally, the lack of data ownership for students hinders their control over their educational records, limiting their ability to securely share or revoke access as needed. These challenges underscore the critical need for a robust and innovative solution, such as our "Transparent Education Data Management using Blockchain" project, to address these pervasive issues and foster trust, security, and efficiency in the management of educational data.

2.2 Problem Statement Definition

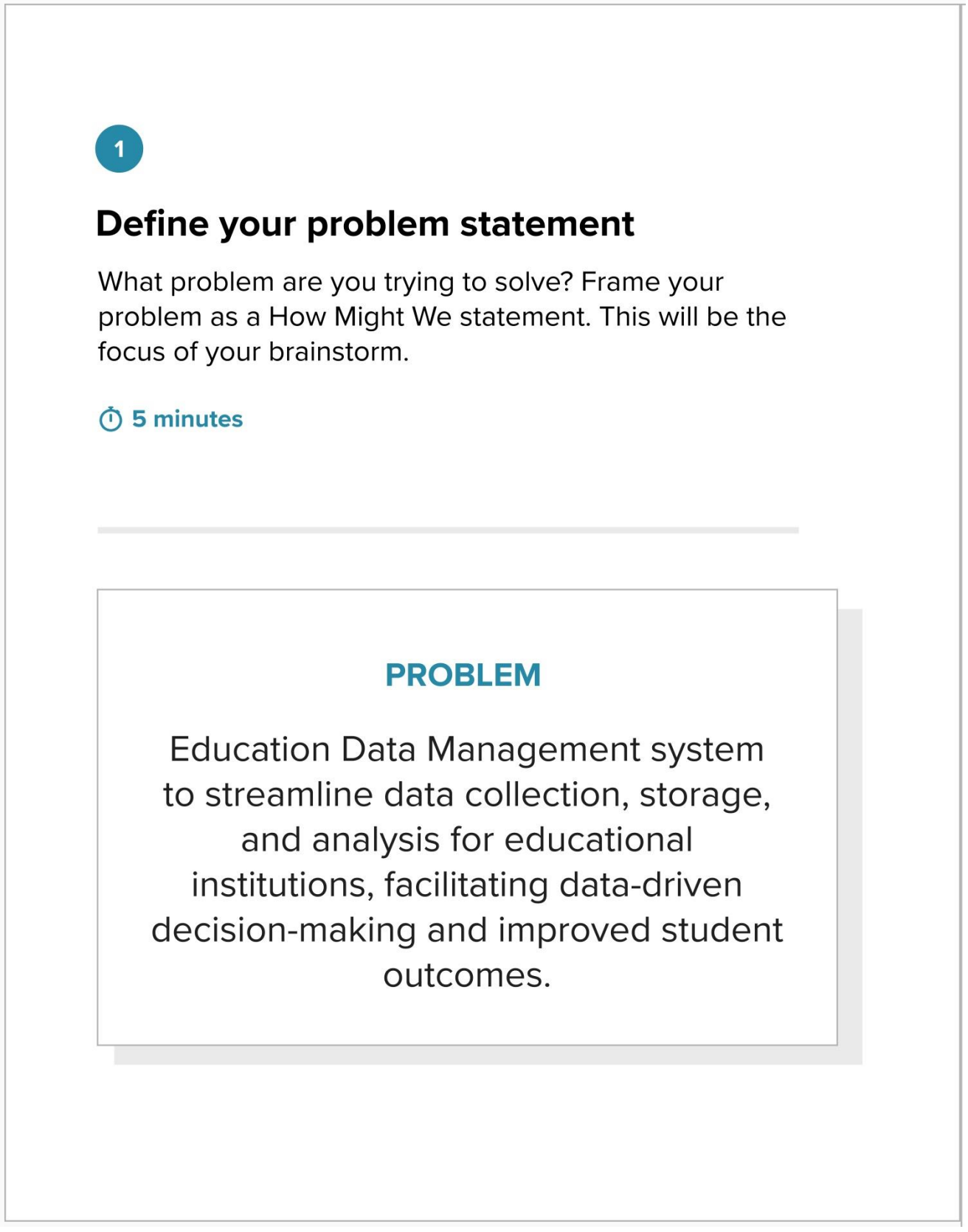
A problem statement is a comprehensive and structured description of a specific issue or challenge that requires attention and resolution. It serves as a foundational element in problem-solving and decision-making processes by articulating the gap between the current state of affairs and the desired state, emphasizing the essential characteristics and ramifications of the problem. An effective problem statement typically identifies the nature of the problem, its scope and impact, and why it is of significant concern. It provides context and clarity, allowing stakeholders and decision-makers to understand the problem's complexity and urgency. In the context of our "Transparent Education Data Management using Blockchain" project, the problem statement might encompass the inadequacies of existing educational data management systems. It could describe how these systems fall short in terms of data security, integrity, and user control, resulting in vulnerabilities, privacy issues, and inefficiencies in the verification of academic credentials. By defining the problem in this manner, it becomes the focal point for the project, guiding efforts to develop an innovative solution to address these pressing concerns and improve the management of educational data.

3. IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation and Brainstorming



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

- Implement real-time performance dashboards for administrators.
- Create a blockchain-based certification verification system.

Person 2

- Develop predictive models for at-risk student identification.
- Design data visualization tools for easy data interpretation.

Person 3

- Build a mobile app for student and parent access.
- Enhance data security and privacy with encryption.

Person 4

- Improve user interface for educator and admin ease of use.
- Conduct user testing for better accessibility and experience.

Person 5

- Develop a recommendation engine for educational resources.
- Provide data-driven decision-making training to schools.

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Create an all-in-one education data management system with real-time dashboards, blockchain-based certification verification, and a web app for secure data access.

Develop predictive models for identifying at-risk students, enhance data visualization tools, and ensure a user-friendly interface with regular testing.

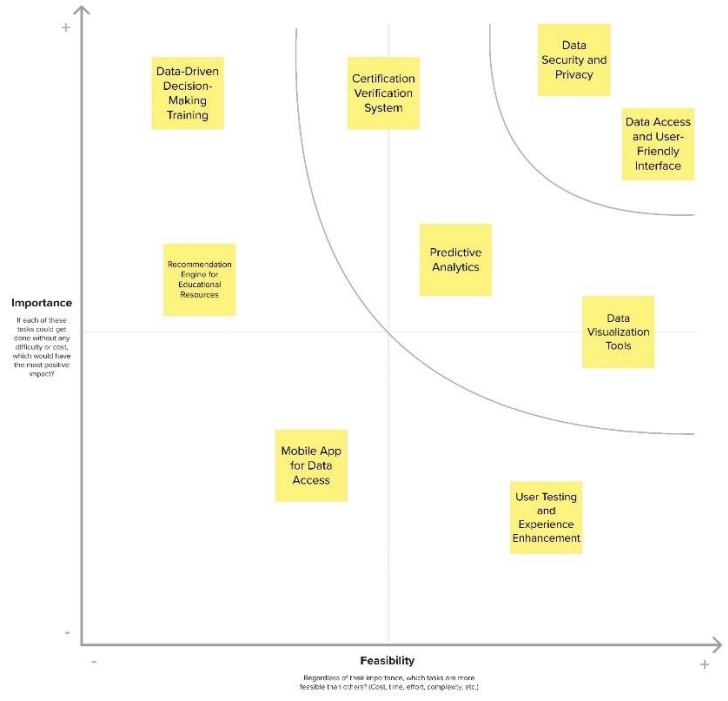
Implement a recommendation engine for educational resources and offer data-driven decision-making training to schools for more informed choices.

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



4. Requirement Analysis

4.1 Functional Requirements

User Registration and Authentication: Implement a user registration system to allow students, educational institutions, and employers to create and manage their accounts securely. Ensure robust authentication mechanisms, such as two-factor authentication, to protect user identities.

Blockchain Integration: Integrate a blockchain network to securely store and manage educational records, ensuring data immutability and decentralization. The blockchain should support smart contracts for automated processes.

Record Submission and Verification: Enable students to submit their educational records to the blockchain system, and provide institutions and employers with the ability to verify these records efficiently. This should include automated verification processes using smart contracts.

Data Ownership and Access Control: Develop a user-friendly interface that allows students to have control over their educational records, granting and revoking access to their data securely. Ensure granular permission settings for sharing data with authorized parties.

Transcript Generation: Create a feature for institutions to generate and issue digital transcripts directly from the system, with the transcript being cryptographically signed to ensure its authenticity.

Data Encryption: Implement robust encryption methods to protect data at rest and during transmission to guarantee data confidentiality.

Audit Trail: Maintain a comprehensive audit trail that records all data access and changes, enhancing transparency and accountability.

User Support and Helpdesk: Establish a user support system, including a helpdesk or customer support feature, to assist users with any issues or inquiries.

Compatibility and Integration: Ensure compatibility with existing education systems and databases, allowing for a smooth transition for educational institutions.

Scalability: Design the system to be scalable to accommodate a growing number of users, institutions, and records.

Reporting and Analytics: Provide reporting and analytics tools to give institutions insights into data access and verification trends.

Compliance with Data Regulations: Ensure that the system complies with data privacy and protection regulations, such as GDPR or HIPAA, depending on the region and type of data involved.

User Training and Documentation: Develop user training materials and documentation to help users navigate the system effectively.

These requirements represent the core functionalities that are fundamental to the success of your blockchain-based transaction management system.

4.2 Non-Functional Requirements

Performance: The system should be highly responsive, with low latency, to ensure quick access to educational records and verification processes. It should support a large number of concurrent users and handle peak loads efficiently.

Scalability: The system should be designed to scale both vertically and horizontally to accommodate the growing number of users and increasing data volumes as the user base expands.

Security: Implement stringent security measures, including encryption, access control, and regular security audits, to protect data from unauthorized access and cyber threats. Ensure compliance with industry-standard security practices.

Reliability: The system should be highly reliable, with minimal downtime, to ensure continuous access to educational records. Implement redundancy and failover mechanisms to minimize disruptions.

Usability: The user interface should be intuitive and user-friendly, requiring minimal training for users to navigate the system effectively. It should also be accessible to users with disabilities.

Compliance: Ensure compliance with relevant data privacy and protection regulations, such as GDPR, HIPAA, or other regional data laws, depending on the scope of the project.

Interoperability: The system should be able to integrate with other education systems and databases, ensuring data can be shared seamlessly with authorized parties.

Data Backup and Recovery: Regularly back up educational records to prevent data loss and ensure quick recovery in case of system failures or data corruption.

Load Testing: Perform load testing to ensure the system can handle heavy usage without performance degradation. This testing should identify the system's maximum capacity.

Audit and Logging: Maintain comprehensive audit logs of all system activities and access to data for accountability and traceability.

Data Retention Policies: Establish data retention and data disposal policies in compliance with relevant regulations, detailing how long data will be stored and how it will be securely deleted.

Mobile Responsiveness: The system should be mobile-responsive, allowing users to access and manage their educational records on various mobile devices.

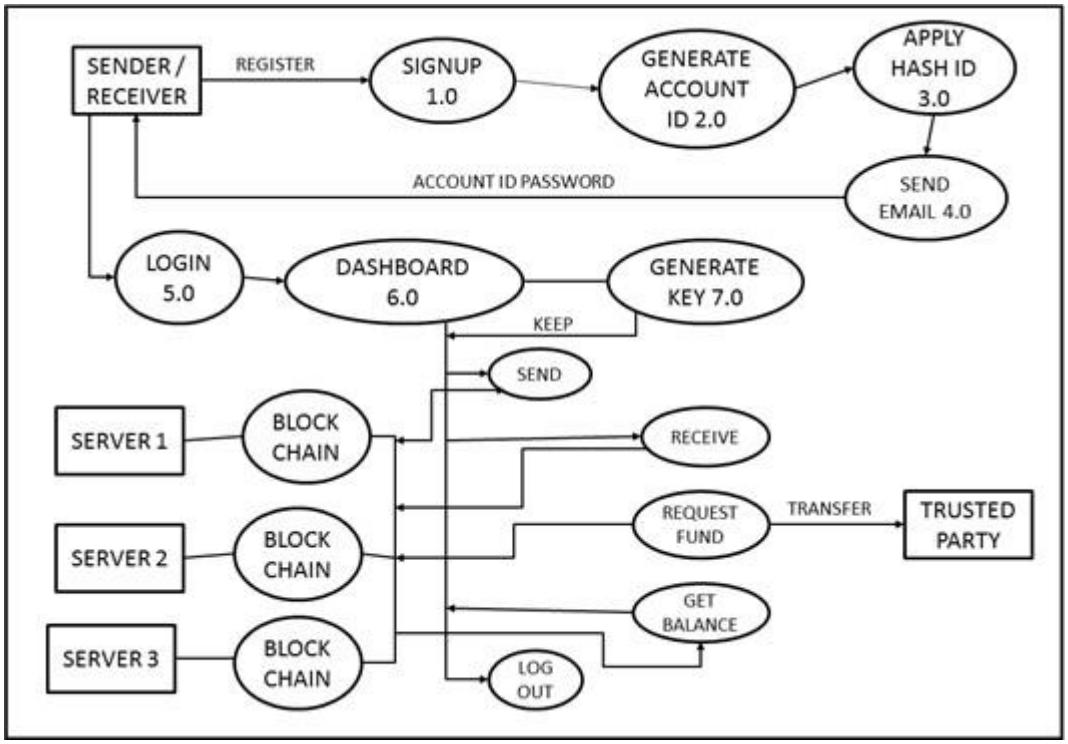
User Support: Offer robust user support with clear response times, providing assistance to users in case they encounter any issues or have inquiries.

Documentation: Provide comprehensive documentation for system administrators, users, and developers, detailing system functionality and maintenance procedures.

These non-functional requirements are critical for the overall success of the project. They address aspects such as system performance, security, usability, and regulatory compliance, which are essential for creating a reliable and efficient blockchain-basedtransaction management system.

5. PROJECT DESIGN

5.1 Data Flow Diagram & User Stories



Transparent Education Data Management In Blockchain

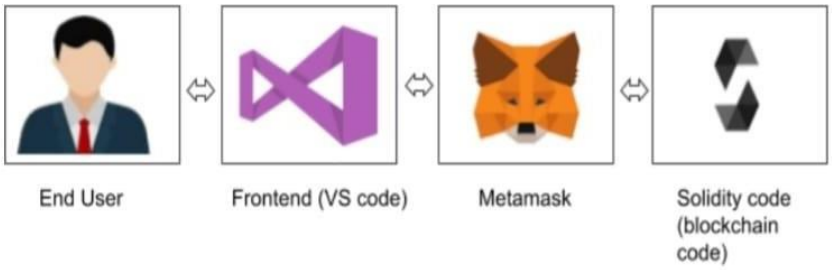
Story 1

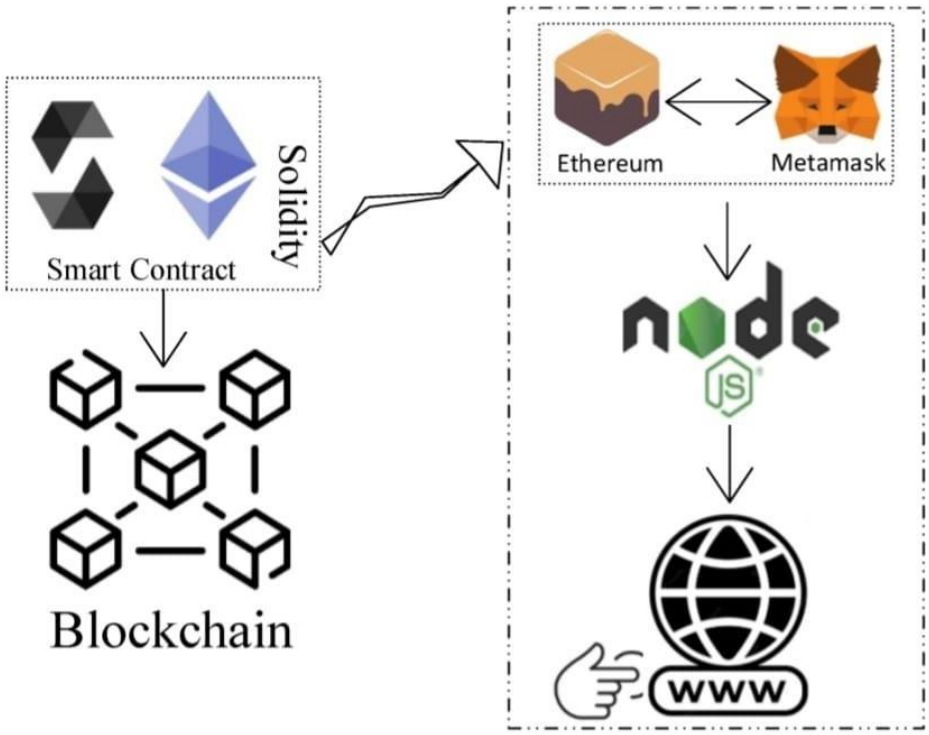
As a high school graduate, Alex is excited about embarking on a higher education journey. He understands the importance of academic records but is equally concerned about their security and authenticity. The conventional way of managing and sharing transcripts feels archaic and insecure. With the introduction of the blockchain-based education data management system, Alex's worries dissipate. He registers on the system effortlessly and uploads his high school transcripts and recommendation letters. These documents, now stored on a tamper-proof blockchain, provide him with an immutable record of his achievements. The true value, however, lies in the control he gains over his data. Alex configures permission settings, ensuring that only the institutions and organizations he authorizes can access his records. This newfound level of control and security not only safeguards his privacy but also simplifies the application process for colleges. It's a revolution in how students manage their academic data, granting them autonomy and providing institutions with trustworthy, readily accessible records.

Story 2

In the heart of University, Dr. Emily, the admissions officer, faces a constant stream of prospective students eager to join the institution. The verification of academic credentials, once a tedious and time-consuming process, now poses a manageable task. Thanks to the blockchain-based education data management system, Dr. Emily can efficiently access and verify student transcripts. With a few keystrokes, she finds the records she needs, each document cryptographically signed to ensure its authenticity. This eliminates any doubts about the veracity of the data, allowing for swift and confident admission decisions. Moreover, the system's audit trail keeps a comprehensive record of the verification process, enhancing transparency and accountability. The system doesn't just save time; it reassures Dr. Emily and the universities that the credentials presented are reliable, making the admissions process smoother and more efficient.

5.2 Solution Architecture

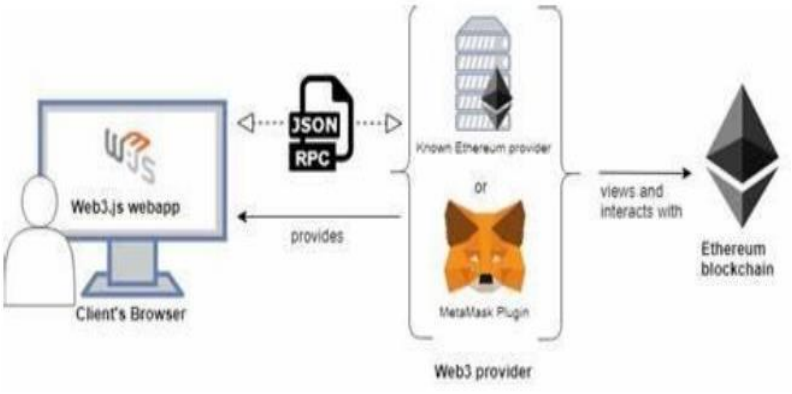




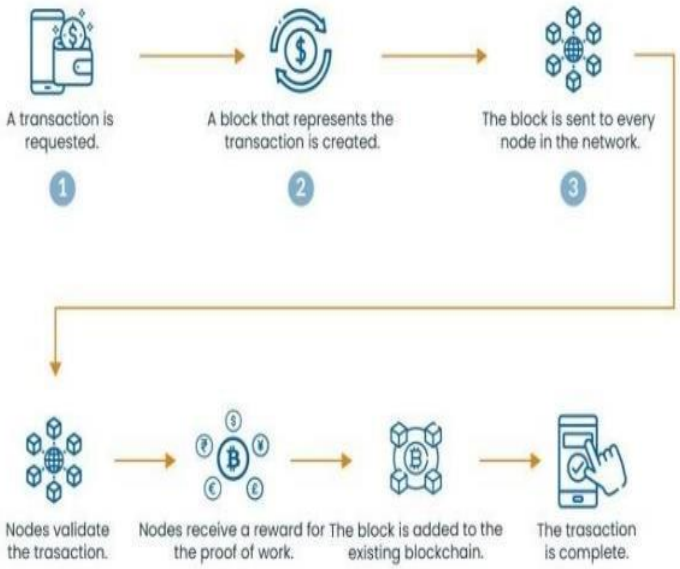
Interaction between web and the Contract

6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture



GENERAL ARCHITECTURE



FLOW OF TRANSACTIONS

6.2 Sprint Planning and Estimation

Sprint planning involves selecting work items from the product backlog and committing to completing them during the upcoming sprint

Sprint 1: Project Initialization and Architecture Design

In the initial sprint, the team will focus on setting clear project objectives, defining goals, and outlining the scope of work. It will include selecting the appropriate blockchain platform, such as Ethereum or Hyperledger, which forms the backbone of the system. Additionally, assembling the development team and creating the system's architectural design are key tasks. The team will set up the development environment and ensure everyone is aligned with the project's direction.

Estimation: This sprint is expected to take approximately 2 weeks to complete.

Sprint 2: User Registration and Authentication

Sprint 2 involves building the foundational components of the system, particularly user registration and authentication. This includes creating user registration functionality and implementing secure authentication mechanisms. The team will design user account management features and develop user-friendly login and registration interfaces for an easy onboarding process.

Estimation: The team anticipates that this sprint will take about 2 weeks to finish.

Sprint 3: Blockchain Integration and Smart Contracts

In this sprint, the focus shifts to the heart of the project – the integration of blockchain technology. The team will integrate the selected blockchain platform with the system and develop smart contracts for crucial functionalities like record submission and verification. Comprehensive testing will ensure the blockchain integration is seamless, and data security, encryption, and privacy measures are in place.

Estimation: This sprint is expected to take approximately 3 weeks to complete.

Sprint 4: Record Submission and Verification Features

Sprint 4 is dedicated to building features that empower students and institutions. The team will create a user-friendly interface for record submission and implement efficient verification processes for institutions. The development of granular permission settings for data access control will be a critical part of this sprint. Rigorous testing will ensure these features work flawlessly.

Estimation: The team anticipates that this sprint will take about 3 weeks to complete.

Sprint 5: User Interface Enhancements and Usability

Sprint 5 is aimed at improving user experience. The team will enhance the user interface, making it more user-friendly, intuitive, and aesthetically pleasing. Mobile responsiveness will be a focus, ensuring that users can access the system on a variety of devices. User testing and feedback collection will be integral to iterative enhancements.

Estimation: The team estimates that this sprint will take approximately 2 weeks to complete.

Sprint 6: Security and Compliance Enhancements

Security and compliance take center stage in this sprint. The team will implement advanced security measures, including encryption and access control, to protect user data. It will ensure compliance with data protection regulations relevant to the project's region, such as GDPR or HIPAA. Security testing and vulnerability assessments will be conducted for added assurance.

Estimation: This sprint is expected to take approximately 3 weeks to complete.

Sprint 7: Performance Optimization and Scalability

Sprint 7 focuses on ensuring the system is both high-performing and scalable. The team will optimize system performance to enhance responsiveness and undertake load testing to ensure that the system can handle high volumes of data and users efficiently.

Estimation: This sprint is expected to take 2 weeks to finish.

Sprint 8: Documentation, Training, and Final Testing

In the final sprint, the team will create comprehensive documentation for users, administrators, and developers to facilitate efficient system usage and maintenance. They will develop training materials to help users navigate the system effectively. The sprint concludes with final testing, including user acceptance testing to ensure the system meets its intended functionality.

Estimation: The team estimates that this sprint will take about 2 weeks to complete.

6.3 Sprint Delivering Schedule

Week 1: Initial Development and Testing

- Focus on the most critical and foundational aspects of the project.
- Prioritize project initialization, blockchain integration, and basic user registration.
- Conduct initial testing to identify and address any major issues or bottlenecks.
- Set up the development environment and configure the selected blockchain platform.

Week 2: Core Functionality and Security

- Concentrate on implementing the core functionality of the system, specifically, record submission, and verification features.
- Enhance system security by implementing access control and encryption measures.
- Continue to test the core features to ensure they are functioning correctly and securely.
- Address any security vulnerabilities or issues identified during testing.

Week 3: Usability and Documentation

- Focus on improving the user interface, making it more user-friendly and mobile-responsive.
- Address usability issues identified in user testing.
- Develop initial documentation for users and administrators.
- Perform a final round of testing and bug fixes.
- Prepare the system for a small-scale pilot or limited user deployment, if feasible.

7. CODING AND SOLUTIONING

7.1 Feature 1

Smart contract (Solidity)

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

contract collegeCertificate {
    address public owner;

    struct Certificate {
        string studentName;
        string courseName;
        uint256 DateOfGraduation;
        uint256 issueDate;
        address issuer;
    }

    uint256 public totalCertificates;
    mapping(uint256 => Certificate) public certificates;

    event CertificateIssued(
        uint256 indexed certificateId,
        string studentName,
        string courseName,
        uint256 issueDate,
        address indexed issuer
    );

    constructor() {
        owner = msg.sender;
    }

    modifier onlyOwner() {
        require(msg.sender == owner, "Only contract owner can call this");
        _;
    }
```

```
    function issueCertificate(
        string memory studentName,
        string memory courseName,
        uint256 _dateOfGraduation,
        uint256 issueDate
    ) external onlyOwner {
        uint256 certificateId = totalCertificates + 1;

        certificates[certificateId] = Certificate({
            studentName: studentName,
            courseName: courseName,
            DateOfGraduation : _dateOfGraduation,
            issueDate: issueDate,
            issuer: msg.sender
        });

        totalCertificates = certificateId;

        emit CertificateIssued(
            certificateId,
            studentName,
            courseName,
            issueDate,
            msg.sender
        );
    }

    function getCertificate(
        uint256 certificateId
    ) external view returns (string memory, string memory, uint256, uint256, address) {
        Certificate memory cert = certificates[certificateId];
        return (cert.studentName, cert.courseName, cert.DateOfGraduation, cert.issueDate, cert.issuer);
    }
}
```

The provided Solidity smart contract, named "collegeCertificate," serves as a tool for managing educational certificates on the Ethereum blockchain. It is designed to offer a transparent and immutable record-keeping system for educational achievements.

At its core, the contract defines a data structure called "Certificate" to represent each certificate. These certificates contain important details such as the student's name, the name of the course or degree obtained, the date of graduation, the date of issuance, and the address of the entity or person issuing the certificate.

The contract maintains two key state variables. First, there's "totalCertificates," which keeps track of the total number of certificates issued. Second, the "certificates" mapping links each certificate's unique ID to its corresponding Certificate struct.

To ensure transparency, the contract logs the issuance of certificates through the "CertificateIssued" event. This event captures relevant information such as the certificate's ID, the student's name, course name, issuance date, and the issuer's address.

The contract includes a "onlyOwner" modifier, allowing certain functions to be accessible only to the contract owner. This is a security measure to ensure that functions related to certificate issuance are controlled by the entity deploying the contract.

The "issueCertificate" function enables the contract owner to issue certificates by providing specific details. It increases the certificate count, creates a Certificate struct, records the certificate's data in the mapping, and emits the "CertificateIssued" event.

Lastly, the "getCertificate" function enables users to retrieve certificate details by specifying the certificate's unique ID. It returns a tuple with relevant information, including the student's name, course name, date of graduation, issuance date, and the issuer's address.

In summary, this contract offers a basic yet functional framework for managing educational certificates securely on the blockchain. It provides transparency, data immutability, and a controlled issuance process, making it suitable for certain educational record-keeping applications. However, in practical applications, the contract's security, scalability, and integration with real-world educational systems would need careful consideration.

Contract ABI (Application Binary Interface):

The abi variable holds the ABI of an Ethereum smart contract. ABIs are essentialfor encoding and decoding function calls and data when interacting with the Ethereum blockchain.

MetaMask Check:

The code first checks whether the MetaMask wallet extension is installed in theuser's browser. If MetaMask is not detected, it displays an alert notifying the user thatMetaMask is not found and provides a link to download it.

Ethers.js Configuration:

It imports the ethers library, which is a popular library for Ethereum development.It creates a provider using Web3Provider, which connects to the user'sMetaMask wallet and provides access to Ethereum. It creates a signer to interact withthe Ethereum blockchain on behalf of the user.It defines an Ethereum contract address and sets up the contract object using ethers.Contract, allowing the JavaScript code tointeract with the contract's functions.In summary, this code is used for interacting with an Ethereum smart contract through MetaMask and ethers.js. It configures the necessary Ethereum provider and signer for communication with the blockchain andsets up a contract object for executing functions and fetching data from the specifiedcontract address using the provided ABI.

8. PERFORMANCE TESTING

8.1 Performance Metrics

Response Time: Measure the time it takes for the system to respond to user requests. This includes actions such as record submission, data access, and verification. Shorter response times indicate a more responsive system.

Throughput: Evaluate the system's capacity to handle concurrent transactions and users. Higher throughput means the system can process a greater number of operations simultaneously.

Latency: Monitor the time delay between a user's request and the system's response. Low latency is crucial for real-time interactions and user satisfaction.

Error Rate: Keep track of the frequency of errors or failed transactions. A low error rate indicates system reliability and data accuracy.

Scalability: Measure the system's ability to scale up to accommodate a growing user base and increased data volume. Scalability is crucial for handling future expansion.

Resource Utilization: Assess how efficiently the system uses resources such as CPU, memory, and storage. Efficient resource utilization helps optimize costs and performance.

Security Incidents: Monitor and report on the number of security incidents, such as unauthorized access attempts, data breaches, or vulnerabilities. A lower incidence of security issues is a key performance indicator.

Uptime and Availability: Measure the system's availability and uptime. A highly available system minimizes downtime and ensures that educational records are accessible when needed.

Transaction Verification Time: Track the time it takes to verify the authenticity of educational records, especially in cases of verification by academic institutions. Faster verification times improve user experience.

Audit Log Analysis: Analyze the audit logs to ensure transparency, traceability, and compliance with data protection regulations. Timely and accurate log analysis is essential for accountability.

User Satisfaction: Collect feedback from users to gauge their satisfaction with the system. User satisfaction surveys or ratings can provide valuable insights into the system's performance from the user's perspective.

Data Storage Efficiency: Monitor the efficiency of data storage to ensure that the system is not wasting resources on redundant or unnecessary data storage.

Concurrent Users: Keep track of the number of concurrent users accessing the system. This metric helps in understanding the system's capacity to serve multiple users simultaneously.

Load Testing Results: Review the results of load testing to understand how the system performs under heavy usage conditions. This data is essential for identifying performance bottlenecks and optimizing the system.

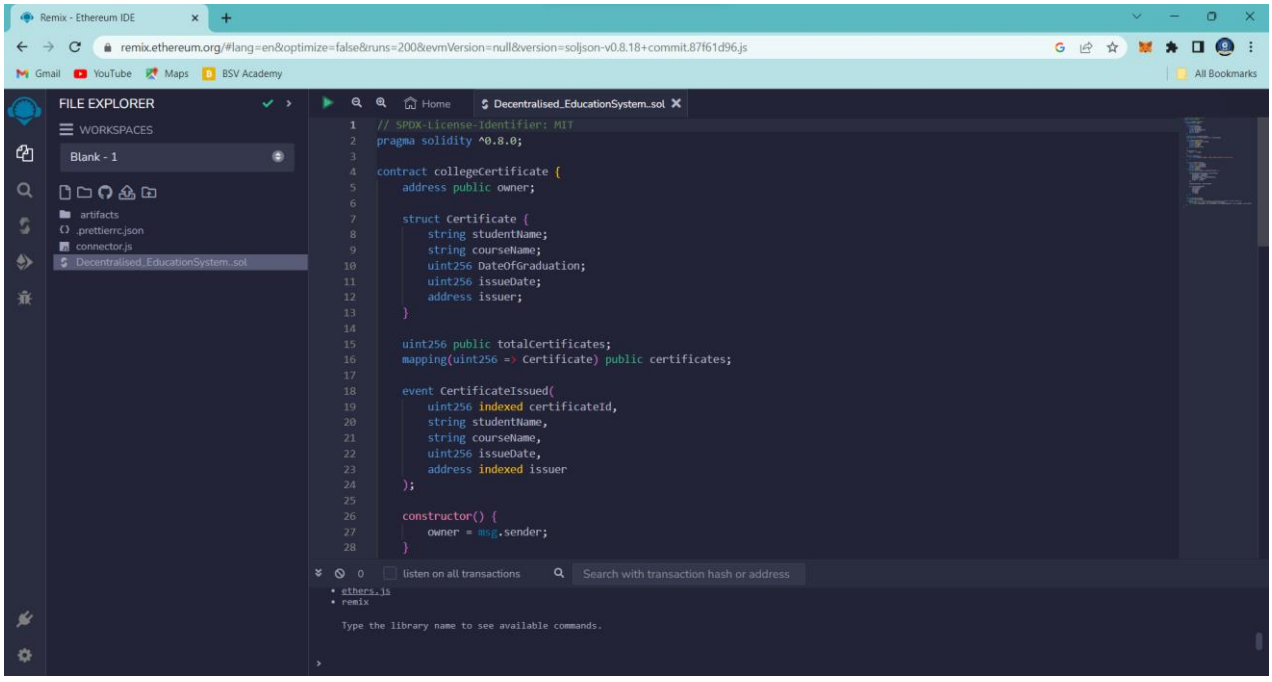
Data Backup and Recovery Time: Measure the time required for data backup and recovery. A shorter recovery time is crucial for minimizing data loss in case of system failures.

Compliance Metrics: Evaluate the system's adherence to data privacy and protection regulations. Ensure that the system complies with relevant laws and industry standards.

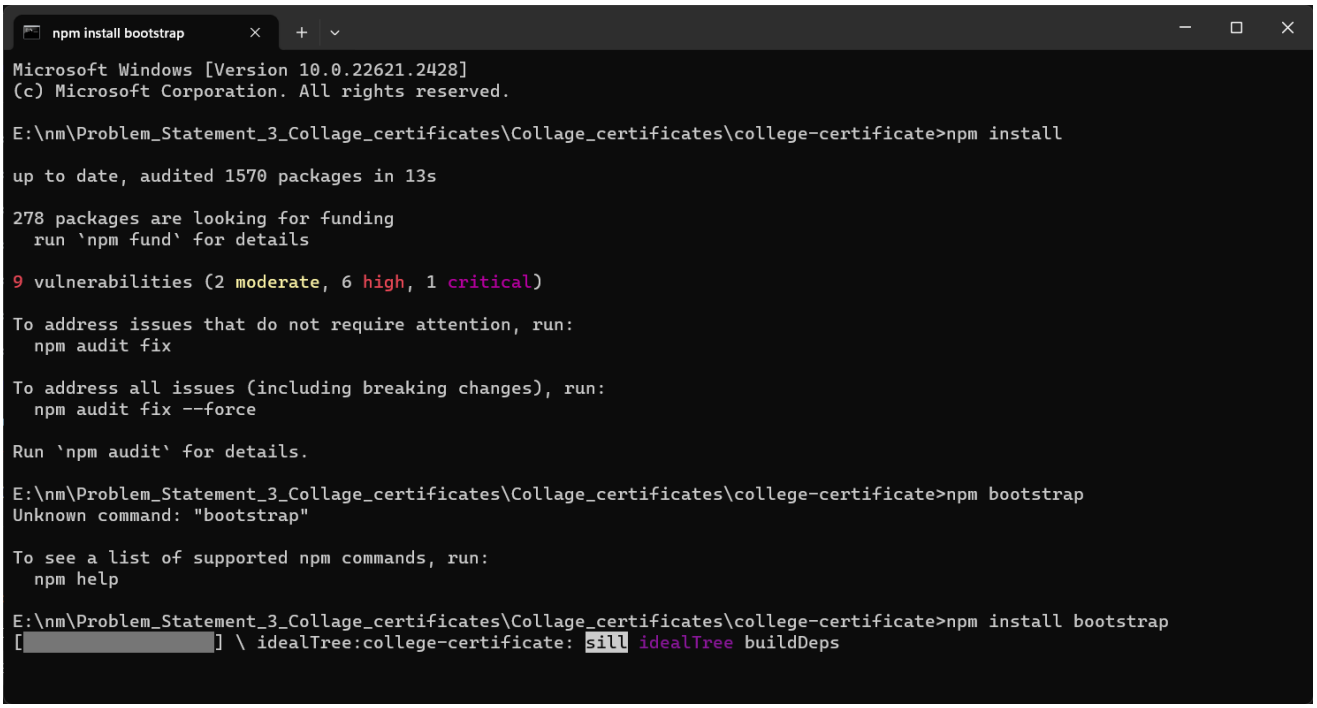
Usage Trends: Analyze usage patterns and trends over time to make informed decisions about system optimization and resource allocation.

9. RESULTS

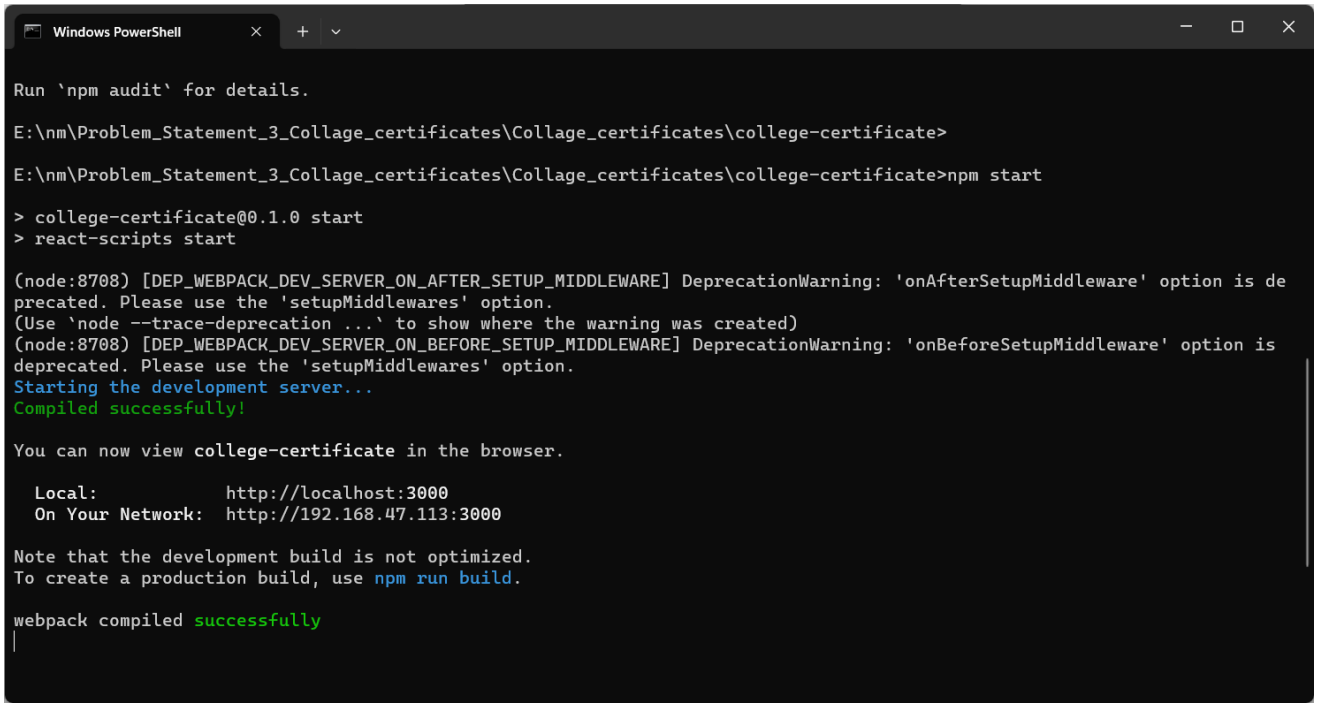
9.1 Output Screenshots



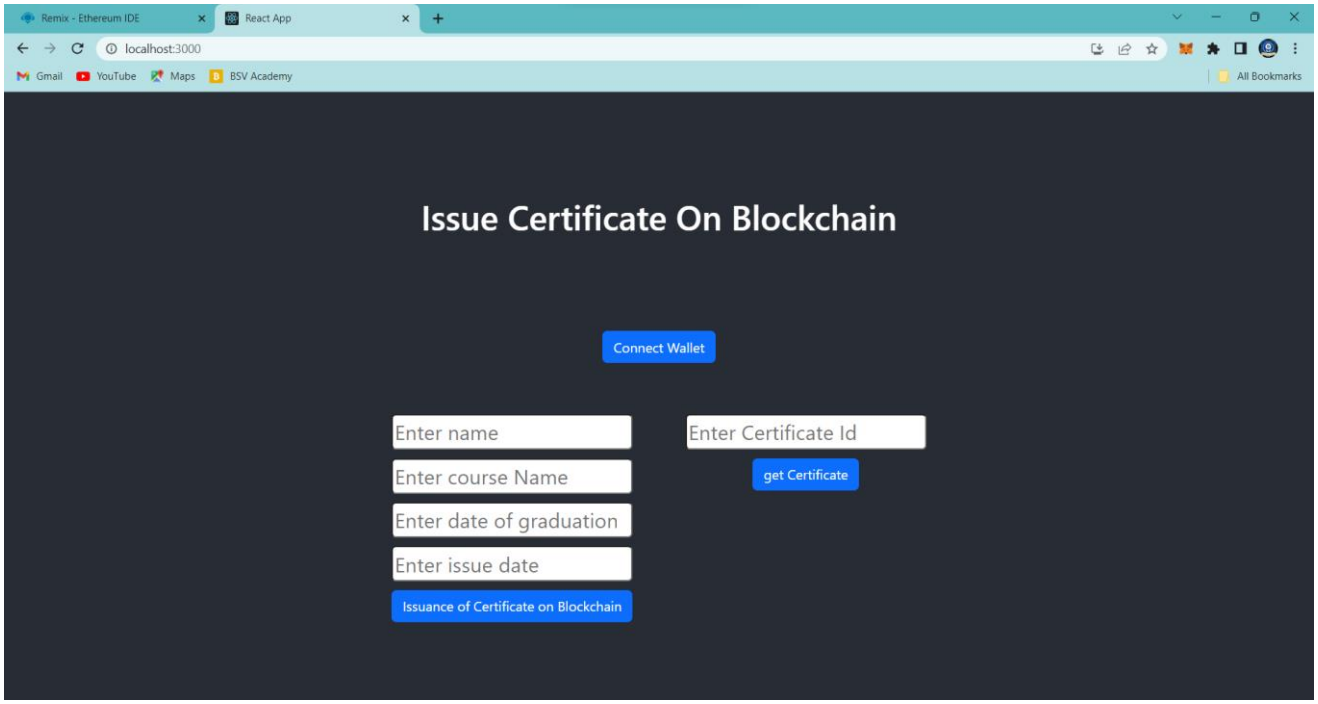
CREATING A SMART CONTRACT



INSTALLING DEPENDENCIES



HOSTING THE SITE LOCALLY



OUTPUT SCREEN

10. ADVANTAGES AND DISADVANTAGES

10.1 Advantages

The "Transparent Education Data Management using Blockchain" project offers several distinct advantages in the realm of educational data management. First and foremost, it provides an unprecedented level of transparency and data security. By leveraging blockchain technology, the project ensures that educational records, including transcripts and certificates, are stored in an immutable and tamper-proof manner. This transparency not only fosters trust but also minimizes the risk of data manipulation, which is critical in the education sector where the accuracy and integrity of records are paramount.

Another significant advantage is the empowerment it provides to both students and academic institutions. Students gain control over their records, deciding who can access them and when. They can easily share their academic achievements with potential employers or other educational institutions, streamlining the application and verification processes. Academic institutions benefit from the efficiency of the system, as it simplifies and accelerates the verification of student credentials. This not only reduces administrative overhead but also enhances the trustworthiness of their admissions process.

Furthermore, the project enhances data security and privacy. Advanced security measures, including encryption and access control, safeguard sensitive educational data, ensuring that it is only accessible to authorized parties. The compliance with data protection regulations adds an extra layer of data security, making the system more appealing to users who are concerned about their privacy.

In conclusion, the "Transparent Education Data Management using Blockchain" project is a game-changer in the education sector. It combines transparency, data security, user empowerment, and efficiency, creating a robust system for the secure and efficient management of educational records. This not only simplifies administrative processes but also provides a trustworthy and secure environment for students to manage and share their academic achievements.

10.2 Disadvantages

The "Transparent Education Data Management using Blockchain" project, while promising, presents several notable disadvantages. Firstly, its technical complexity can be a barrier to adoption. The intricacies of blockchain technology require a significant level of expertise, making it challenging for educational institutions and users to grasp and effectively navigate. This complexity could result in a steep learning curve and hinder widespread acceptance, particularly among institutions and individuals less familiar with blockchain systems.

Secondly, scalability poses a significant challenge. As the user base and data volume expand, blockchain networks can experience congestion, leading to slower transaction processing and increased operational costs. Achieving efficient scalability in blockchain systems is complex, and not all platforms are well-suited for managing extensive educational data, potentially impacting real-time functionality and system performance. These scalability issues need careful consideration to ensure the system can accommodate the growth and demands of educational data management effectively.

11. CONCLUSION

In conclusion, the "Transparent Education Data Management using Blockchain" project represents a promising leap forward in the education sector, offering several advantages such as enhanced data security, transparency, user empowerment, and administrative efficiency. However, it is important to recognize the project's associated disadvantages and challenges.

One of the significant drawbacks lies in the technical complexity of blockchain technology. Implementing and maintaining such a system requires a high level of expertise and understanding. For educational institutions and users unfamiliar with blockchain, this could present a formidable barrier. The steep learning curve may slow down the adoption of this innovative solution.

Scalability is another concern. As the user base and data volume grow, blockchain networks can experience congestion, potentially leading to delays and increased costs. Addressing scalability issues efficiently is a complex task, especially when managing extensive educational data in real-time scenarios.

Furthermore, the immutability of blockchain data poses challenges in correcting errors or updating records. Once data is recorded, it becomes nearly irreversible, which can be problematic in situations where data modifications are necessary.

Energy consumption is a growing concern as well, with some blockchain networks requiring substantial computational power and energy resources, contributing to environmental sustainability concerns.

Interoperability with existing systems and databases is another challenge, as educational institutions often already have established systems for managing student records.

Lastly, navigating the evolving regulatory landscape related to blockchain in education, including data protection and privacy regulations, can be complex.

In light of these disadvantages, successful implementation of this project will require careful planning, specialized expertise, and adaptation to overcome the challenges. By addressing these issues, the project has the potential to revolutionize education data management, offering a secure, transparent, and efficient solution that empowers both educational institutions and students while enhancing trust and data integrity in the sector. However, the road to realizing these benefits requires a balanced approach that acknowledges and mitigates the associated challenges.

12. FUTURE SCOPE

The "Transparent Education Data Management using Blockchain" project holds significant potential for future expansion and development. Here are some key future scope areas:

Wider Adoption: As blockchain technology becomes more mainstream and better understood, the project has the potential for widespread adoption across educational institutions, not only at the university level but also in K-12 education. Governments and educational authorities could explore its application for managing standardized testing and certification records.

Credential Verification Services: The system can evolve into a comprehensive credential verification service. It could offer third-party verification services for employers, background check companies, and other institutions, streamlining the verification of academic records.

International Credential Recognition: The project could address the challenge of international credential recognition. By integrating with international education systems and organizations, it can facilitate the recognition of academic qualifications across borders.

Smart Contracts for Enrollment: The use of smart contracts could be expanded to automate the enrollment process. For instance, students could set conditions for their enrollment in certain courses or programs, and the smart contract could execute these conditions automatically.

Integration with eLearning Platforms: Integration with eLearning platforms and Massive Open Online Courses (MOOCs) could enable the direct recording of course completion and achievement on the blockchain, further enhancing the transparency and security of online learning.

Mobile Applications: Developing user-friendly mobile applications can make it even more accessible, allowing students to manage their records on the go and enabling institutions to verify credentials more conveniently.

Blockchain Interoperability: Exploring compatibility with other blockchains and networks, such as Ethereum, Hyperledger, or Polkadot, can improve scalability and interconnectivity, enabling broader usage and data sharing.

Digital Identity Management: Integration with digital identity systems can enhance security and streamline user access to their records. Users could have a single, secure digital identity for all their educational credentials.

Enhanced Analytics: The blockchain's immutability allows for the secure collection of data for educational analytics. This data can be used to improve learning outcomes, identify trends, and personalize educational experiences.

Data Portability: Enabling users to easily transfer their records between different institutions or educational systems can promote data portability and empower students to pursue their educational goals seamlessly.

Blockchain as a Service (BaaS): Offering the project as a BaaS solution to educational institutions, allowing them to implement the technology without the complexities of blockchain development.

Research and Development: Continuous research and development efforts can focus on optimizing the system, addressing scalability, energy efficiency, and interoperability, and exploring new blockchain technologies.

In summary, the future scope for the "Transparent Education Data Management using Blockchain" project is promising, with the potential to revolutionize the way educational records are managed and verified. By addressing current challenges and exploring new avenues, this project can contribute to more transparent, efficient, and secure educational systems on a global scale.

13. APPENDIX

Source code

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
contract collegeCertificate {
    address public owner;
    struct Certificate {
        string studentName;
        string courseName;
        uint256 DateOfGraduation;
        uint256 issueDate;
        address issuer;
    }
    uint256 public totalCertificates;
    mapping(uint256 => Certificate) public
certificates;
    event CertificateIssued(
        uint256 indexed certificateId,
        string studentName,
        string courseName,
        uint256 issueDate,
        address indexed issuer
    );

    constructor() {
        owner = msg.sender;
    }
    modifier onlyOwner() {
```

```
        require(msg.sender == owner, "Only
contract owner can call this");
        _;
    }
    function issueCertificate(
        string memory studentName,
        string memory courseName,
        uint256 _dateOfGraduation,
        uint256 issueDate
    ) external onlyOwner {
        uint256 certificateId = totalCertificates
+ 1;
        certificates[certificateId] =
Certificate({
            studentName: studentName,
            courseName: courseName,
            DateOfGraduation :
_dateOfGraduation,
            issueDate: issueDate,
            issuer: msg.sender
        });
        totalCertificates = certificateId;
        emit CertificateIssued(
            certificateId,
            studentName,
            courseName,
            issueDate,
            msg.sender
        );
```

```
    }
    function getCertificate(
        uint256 certificateId
    ) external view returns (string memory,
string memory, uint256, uint256, address) {
        Certificate memory cert =
certificates[certificateId];
        return (cert.studentName,
cert.courseName, cert.DateOfGraduation,
cert.issueDate, cert.issuer);
    }
}
```

Connector.js

```
const { ethers } = require('ethers');
const abi = [
    {
        inputs: [],
        stateMutability: 'nonpayable',
        type: 'constructor',
    },
    {
        anonymous: false,
        inputs: [
            {
                indexed: true,
                internalType: 'uint256',
                name: 'certificateId',
                type: 'uint256',
            },
            {
                indexed: false,
                internalType: 'string',
                name: 'studentName',
```

```
            type: 'string',
        },
        {
            indexed: false,
            internalType: 'string',
            name: 'courseName',
            type: 'string',
        },
        {
            indexed: false,
            internalType: 'uint256',
            name: 'issueDate',
            type: 'uint256',
        },
        {
            indexed: true,
            internalType: 'address',
            name: 'issuer',
            type: 'address',
        },
    ],
    name: 'CertificateIssued',
    type: 'event',
},
{
    inputs: [
        {
            internalType: 'string',
            name: 'studentName',
            type: 'string',
        },
        {
            internalType: 'string',
            name: 'courseName',
            type: 'string',
        },
        {
```

```
            internalType: 'uint256',
            name: '_dateOfGraduation',
            type: 'uint256',
        },
        {
            internalType: 'uint256',
            name: 'issueDate',
            type: 'uint256',
        },
    ],
    name: 'issueCertificate',
    outputs: [],
    stateMutability: 'nonpayable',
    type: 'function',
},
{
    inputs: [
        {
            internalType: 'uint256',
            name: '',
            type: 'uint256',
        },
    ],
    name: 'certificates',
    outputs: [
        {
            internalType: 'string',
            name: 'studentName',
            type: 'string',
        },
        {
            internalType: 'string',
            name: 'courseName',
            type: 'string',
        },
        {
            internalType: 'uint256',
```

```
            name: 'DateOfGraduation',
            type: 'uint256',
        },
        {
            internalType: 'uint256',
            name: 'issueDate',
            type: 'uint256',
        },
        {
            internalType: 'address',
            name: 'issuer',
            type: 'address',
        },
    ],
    stateMutability: 'view',
    type: 'function',
},
{
    inputs: [
        {
            internalType: 'uint256',
            name: 'certificateId',
            type: 'uint256',
        },
    ],
    name: 'getCertificate',
    outputs: [
        {
            internalType: 'string',
            name: '',
            type: 'string',
        },
        {
            internalType: 'string',
            name: '',
            type: 'string',
        },
```

```
        {
            internalType: 'uint256',
            name: '',
            type: 'uint256',
        },
        {
            internalType: 'uint256',
            name: '',
            type: 'uint256',
        },
        {
            internalType: 'address',
            name: '',
            type: 'address',
        },
    ],
    stateMutability: 'view',
    type: 'function',
},
{
    inputs: [],
    name: 'owner',
    outputs: [
        {
            internalType: 'address',
            name: '',
            type: 'address',
        },
    ],
    stateMutability: 'view',
    type: 'function',
},
{
    inputs: [],
    name: 'totalCertificates',
    outputs: [
        {
```

```
            internalType: 'uint256',
            name: '',
            type: 'uint256',
          },
        ],
        stateMutability: 'view',
        type: 'function',
    },
];
if (!window.ethereum) {
    alert('Meta Mask Not Found');
    window.open('https://metamask.io/download/');
}

export const provider = new ethers.providers.Web3Provider(window.ethereum);
export const signer = provider.getSigner();
export const address = '0xd8b934580fcE35a11B58C6D7E468a2833fa8';
export const contract = new ethers.Contract(address, abi, signer);
```

Home.js

```
import React, { useState } from 'react';
import { Button, Container, Row, Col } from 'react-bootstrap';
import 'bootstrap/dist/css/bootstrap.min.css';
import { contract } from './connector';
function Home() {
    const [name, setName] = useState('');
    const [courseName, setCourseName] = useState('');
    const [date, setDate] = useState('');
    const [issueDate, setIssueDate] = useState('');
    const [CertificateId, setCertificateId] = useState('');
    const [CertificateData, setCertificateData] = useState('');
    const [Wallet, setWallet] = useState('');

    const handleName = e => {
        setName(e.target.value);
    };
    const handleCourseName = e => {
```

```
    setCourseName(e.target.value);
};
const handleDate = e => {
    setDate(e.target.value);
};
const handleIssueDate = e => {
    setIssueDate(e.target.value);
};
const handleCertificateId = e => {
    setCertificateId(e.target.value);
};

const issueCertificate = async () => {
    try {
        let tx = await contract.issueCertificate(
            name,
            courseName,
            date,
            issueDate
        );
        let txWait = await tx.wait();
        console.log(txWait);
        alert(txWait.transactionHash);
    } catch (error) {
        alert(error);
    }
};

const getCertificate = async () => {
    try {
        let tx = await contract.getCertificate(CertificateId);
        let arr = [];
        tx.map(e => {
            arr.push(e.toString());
            console.log('Certificate Info: ', e);
        });
```

```
        setCertificateData(arr);
    } catch (error) {
        alert(error);
        console.log('Certificate not fetched');
    }
};

const handleWallet = async () => {
    if (!window.ethereum) {
        return alert('please install metamask');
    }

    const addr = await window.ethereum.request({
        method: 'eth_requestAccounts',
    });

    setWallet(addr[0]);
};

return (
    <div>
        <h1 style={{ marginTop: '30px', marginBottom: '80px' }}>
            Issue Certificate On Blockchain
        </h1>
        {!Wallet ? (
            <Button
                onClick={handleWallet}
                style={{ marginTop: '30px', marginBottom: '50px' }}
            >
                Connect Wallet{' '}
            </Button>
        ) : (
            <p
                style={{
                    width: '250px',
                    height: '50px',
```

```
            margin: 'auto',
            marginBottom: '50px',
            border: '2px solid #2096f3',
          }}
        >
          {Wallet.slice(0, 6)}....{Wallet.slice(-6)}
        </p>
      )}
      <Container style={{ display: 'flex' }}>
        <Row style={{ marginRight: '50px' }}>
          <Col>
            <div>
              <input
                style={{
                  marginTop: '10px',
                  borderRadius: '5px',
                }}
                onChange={handleName}
                type='string'
                placeholder='Enter name'
                value={name}
              />{' '}
              <br />
              <input
                style={{
                  marginTop: '10px',
                  borderRadius: '5px',
                }}
                onChange={handleCourseName}
                type='string'
                placeholder='Enter course Name'
                value={courseName}
              />{' '}
              <br />
              <input
                style={{
```

```
                marginTop: '10px',
                borderRadius: '5px',
              }}
              onChange={handleDate}
              type='number'
              placeholder='Enter date of graduation'
              value={date}
            />
            <br />
            <input
              style={{
                marginTop: '10px',
                borderRadius: '5px',
              }}
              onChange={handleIssueDate}
              type='number'
              placeholder='Enter issue date'
              value={issueDate}
            />{' '}
            <br />
            <Button
              onClick={issueCertificate}
              style={{ marginTop: '10px' }}
              variant='primary'
            >
              Issuance of Certificate on Blockchain
            </Button>
          </div>
        </Col>
      </Row>
      <Row>
        <Col>
          <div>
            {/* <label>Get Certificate</label><br /> */}
            <input
              style={{
```

```
                        marginTop: '10px',
                        borderRadius: '5px',
                    }}
                    onChange={handleCertificateId}
                    type='number'
                    placeholder='Enter Certificate Id'
                    value={CertificateId}
                />{' '}
                <br />
                <Button
                    onClick={getCertificate}
                    style={{ marginTop: '10px' }}
                    variant='primary'
                >
                    get Certificate
                </Button>
                {CertificateData ? (
                    CertificateData.map(e => <p>{e}</p>)
                ) : (
                    <p></p>
                )}
            </div>
        </Col>
    </Row>
</Container>
</div>
);
}

export default Home;
```

App.js

```
import logo from './logo.svg';
import './App.css';
import Home from './Page/Home'
function App() {
  return (
    <div className="App">
      <header className="App-header">
        <Home />
      </header>
    </div>
  );
}

export default App;
```

Index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
reportWebVitals();
```

GITHUB LINK : https://github.com/Nandha2002/NM-Block-Chain.git

DEMO VIDEO LINK :

https://drive.google.com/file/d/1nDXSITCDgtwRexaz9WsTtKHtxMolurv6/view?usp=share_link