CENTRAL BANK SMART CONTRACT

PROJECT REPORT

SUBMITTED BY		**TEAM ID: NM2023TMID04519**
NATHEESH	-963520106033	MENTOR: Mrs. BABY SOLA
JOHN TENI JIO	-963520106026	SPOC: Dr. BHARADWAJ
RAHUL BRIGHT PRINCE	-963520106035	
REXLIN NAYAGAM	-963520106303	

In the partial fulfillment of the requirements for the award of a degree of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

STELLA MARY'S COLLEGE OF ENGINEERING

ARUTHENGANVILAI, KALLUKATTI JUNCTION

AZHIKKAL (PO), KANYAKUMARI-629202.

2023 – 2024 (ODD)

CONTENT

1. Introduction

Block chain is a revolutionary technology that has fundamentally transformed the way we think about data, transactions, and trust in the digital age. It emerged as the underlying technology for crypto currencies like Bit coin, but its applications extend far beyond digital currencies. At its core, block chain is a decentralized, distributed ledger technology that offers a secure and transparent way to record and verify transactions and data.

In a traditional centralized system, a single entity, such as a bank or a government, maintains a central ledger to record and verify transactions. In contrast, block chain operates as a decentralized ledger shared across a network of computers, known as nodes. Each node stores a copy of the block chain, and the system uses a consensus mechanism to ensure that all copies of the ledger remain in sync and accurate.

There is no central authority or intermediary in control of the block chain. This decentralized nature makes it resistant to censorship and tampering. All transactions and data recorded on the block chain are visible to all participants in the network. This transparency enhances trust and accountability. Block chain employs cryptographic techniques to secure data and transactions. Once a block of data is added to the chain, it is virtually impossible to alter, ensuring the integrity of the ledger.

Block chain technology continues to evolve, and its potential applications are continually expanding. Its decentralized, secure, and transparent nature makes it a powerful tool for industries and sectors seeking to enhance trust and efficiency in a digital world. In this paper we introduce about the Central Bank Smart Contract in block chain.

1.1 Project Overview

Central bank smart contracts are a new and emerging technology with the potential to revolutionize the way central banks operate. Central bank smart contracts can be used to automate and streamline central bank operations, such as currency issuance, payments processing, and monetary policy implementation. They can also be used to create more transparent and accountable central bank systems.

Despite these challenges, there is a growing interest in central bank smart contracts. A number of central banks around the world are exploring the potential uses of central bank smart contracts, and some are already conducting pilot projects.

Develop a prototype central bank smart contract platform that addresses the challenges of scalability, performance, security, privacy, and interoperability. Work with central banks to identify and implement specific use cases for central bank smart contracts. Engage with regulators to develop clear and comprehensive regulation for central bank smart contracts.

The project is expected to take three years to complete. The first year will be focused on developing the prototype central bank smart contract platform. The second year will be focused on working with central banks to identify and implement specific use cases. The third year will be focused on engaging with regulators and preparing for the wider adoption of central bank smart contracts.

1.2 Purpose

The purpose of a central bank smart contract is to automate and streamline central bank operations, create more transparent and accountable central bank systems, and enable new and innovative financial services.

Central bank smart contracts can be used to automate the issuance of new currency and to manage the supply of currency in circulation. This can help to make central bank currency management more efficient and transparent. Central bank smart contracts can be used to process payments between financial institutions and individuals. This can help to make payments faster, cheaper, and more secure.

Central bank smart contracts can be used to provide new financial services, such as central bank digital currencies (CBDCs) and programmable money. CBDCs are digital currencies that are issued and backed by central banks. Programmable money is money that can be programmed to perform specific tasks, such as only being used to purchase certain goods or services.

Central bank smart contracts can be used to implement monetary policy tools, such as interest rate changes and quantitative easing. This can help central banks to maintain price stability and promote economic growth. Central bank smart contracts are still under development, but they have the potential to revolutionize the way central banks operate and to provide new and innovative financial services to the public.

2. Literature Survey

2.1 Existing Problem

Central bank smart contracts need to be able to handle a high volume of transactions without becoming congested or expensive to use. Central bank smart contracts need to be extremely secure to protect against hacking and other attacks. Central bank smart contracts need to protect the privacy of users' financial data. Central bank smart contracts need to be able to interoperate with existing financial systems and infrastructure. There is a need for clear and comprehensive regulation of central bank smart contracts.

In addition to these general problems, there are also some specific challenges that need to be addressed before central bank smart contracts can be widely adopted. For example, central banks need to develop the expertise to manage and operate central bank smart contracts. They also need to coordinate with other central banks and financial regulators to ensure that central bank smart contracts are implemented in a safe and sound manner.

Despite these challenges, central banks are increasingly recognizing the potential benefits of central bank smart contracts. A number of central banks around the world are exploring the potential uses of central bank smart contracts, and some are already conducting pilot projects.

2.2 References

Academic databases like IEEE Xplore, ACM Digital Library, or Google Scholar for peer-reviewed articles and research papers on biometric security in voting systems.

Government election commissions or relevant agencies often publish reports and guidelines on election security, including biometric authentication.

Alaguvel R., Gnanavel G., Jagadhambal K. – "Biometrics using Electronic Voting System with Embedded Security", pp. 1065, 2013.

O.M. Olaniyan, T. Mapayi & S.A. Adejumo – "A Proposed Multiple Scan Biometric-Based System for Electronic Voting", African Journal Comp. & ICT Volume 4. No. 2. Issue 1pp. 12, 2011.

Kashif H.M., Dileep Kumar and Syed Muhammad Usman, “Next Generation A Secure E-Voting System Based On Biometric Fingerprint Method” 2011 International Conference on Information and Intelligent Computing IPCSIT vol.18 (2011) pp .26-27.

OASIS Election & Voter Services Technical Committee – “Requirements for common data formats and standards for e-Voting”, NIST Paper. 18 August 2009 (Retrieved October 10, 2014).

Government election commissions or relevant agencies often publish reports and guidelines on election security, including biometric authentication.

Look for news articles and reports from reputable news outlets that discuss the implementation and challenges of biometric security in voting systems.

Organizations specializing in election technology and security often publish whitepapers and case studies on the subject.

De Giusti A., Feierherd G., Pesado P., Depetris B. “Una aproximación a los requerimientos del software de voto electrónico de Argentina”. Congreso Argentino de Ciencias de la Computación. 2004.

Tula M. “Voto Electrónico”. Ariel Ciencias Políticas. 2005.

Cantijoch Cunill M. “El voto electrónico ¿Un temor justificado?”. Revista TEXTOS de la CiberSociedad, 7. http://www.cibersociedad.net. 2005.

Arsaute G. A., Tutores: Nasisi Óscar Herminio M. M. “Reconocimiento de características en huellas dactilares para la identificación humana”. Universidad Nacional de San Juan. Facultad de Ingeniería. Instituto de Automática. 1997.

Beavan Colin. “Huellas dactilares. Los orígenes de la dactiloscopía”. Ed. Alba. 1990.

Arrieta A., Marín J., Sánchez L. G., Romero L., Sánchez L. A., Batista V. “Gestión y Reconocimiento Óptico de los Puntos Característicos de Imágenes de Huellas Dactilares”. Universidad de Salamanca.

Reid P. “Biometrics for Network Security”. Prentice Hall. 2004.

Chirillo J. y otros. "Implementing Biometric Security". Wiley Publishing. 2003.

2.3 Problem Statement Definition

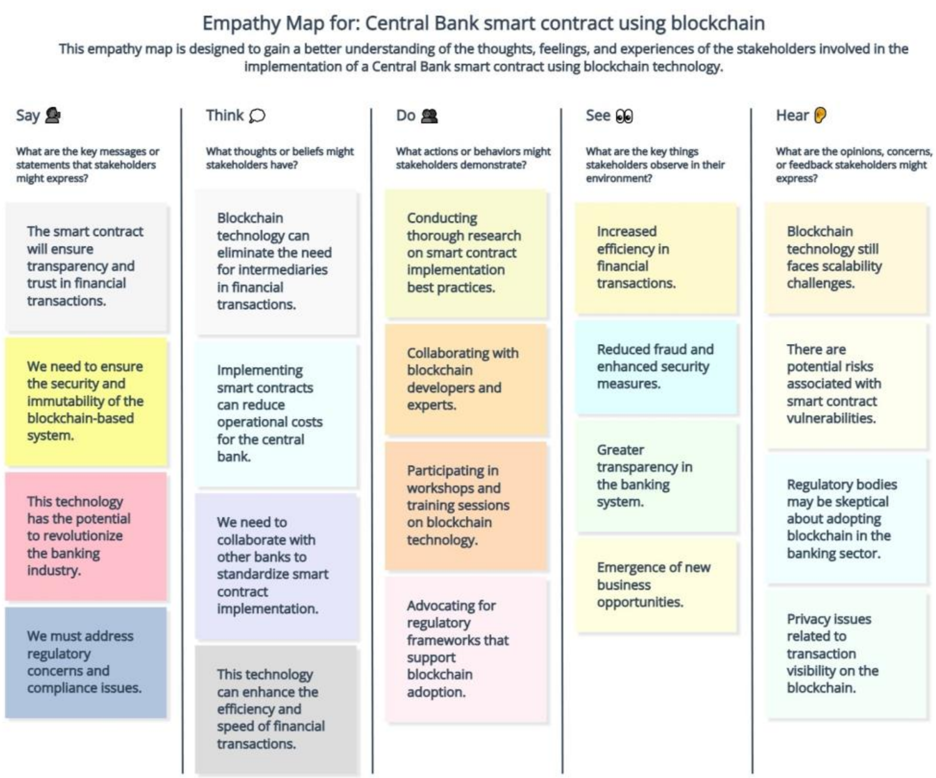
Central bank smart contracts have the potential to revolutionize the way central banks operate, but there are a number of challenges that need to be addressed before they can be widely adopted. These challenges include scalability, performance, security, privacy, interoperability, and regulation.

This problem statement is comprehensive because it identifies all of the key challenges that need to be addressed before central bank smart contracts can be widely adopted. It is also specific because it provides details about each of the challenges, such as the fact that central bank smart contracts need to be scalable enough to handle a high volume of transactions and secure enough to protect against hacking and other attacks.

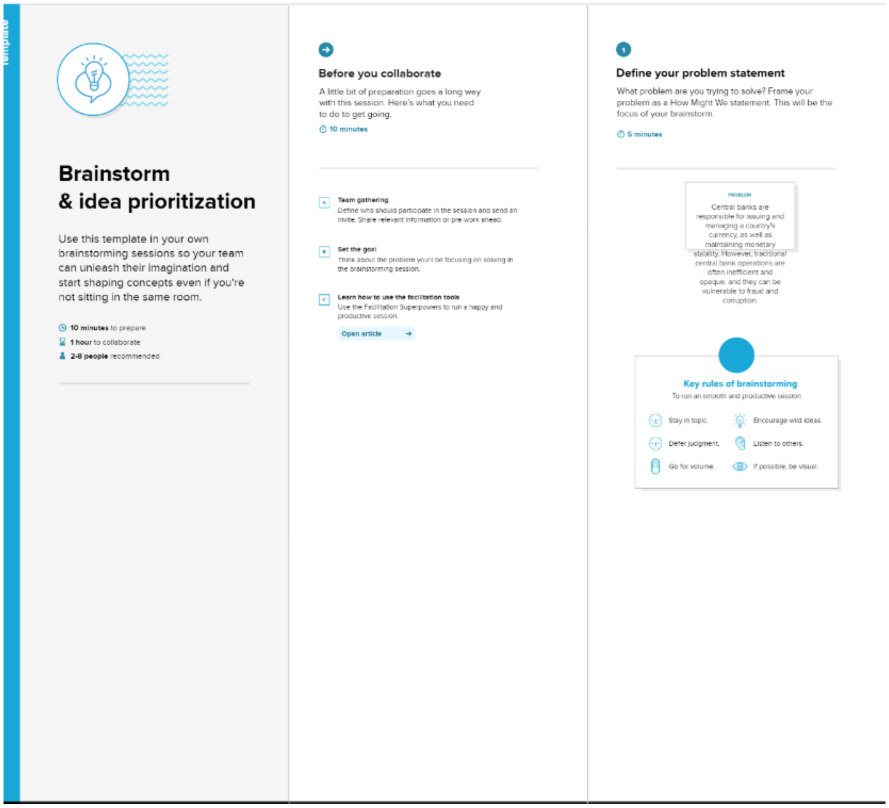
The problem statement is also timely because it is relevant to the current state of central bank smart contract development. Central banks are increasingly recognizing the potential benefits of central bank smart contracts, but they are also aware of the challenges that need to be addressed

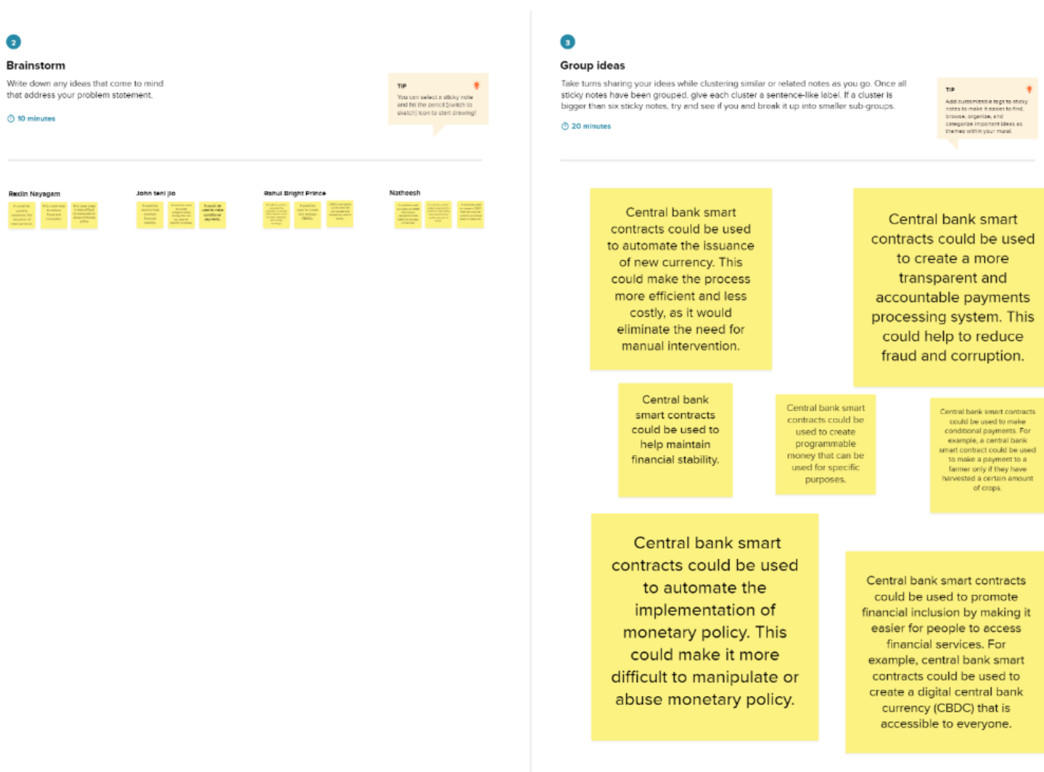
3. Ideation &Proposed Solution

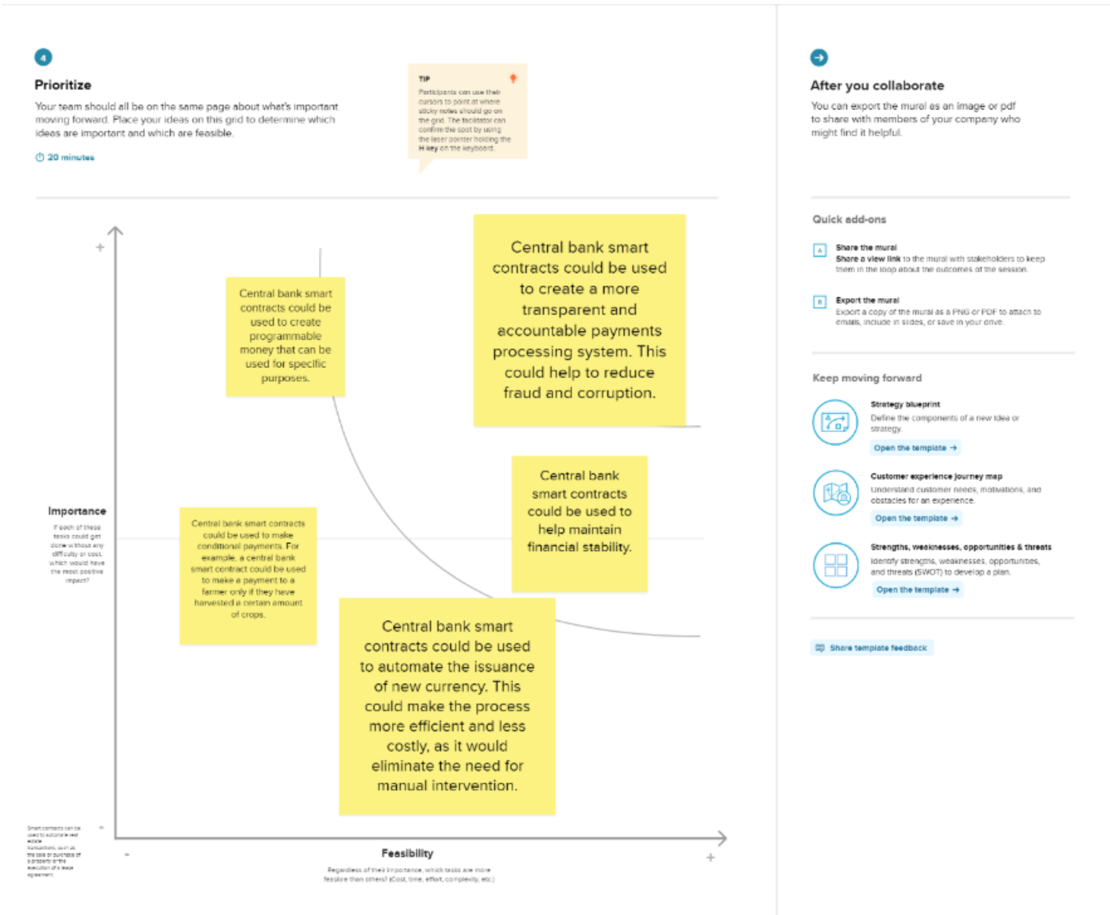
3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming







4. Requirement Analysis

4.1 Functional Requirements

The functional requirements for central bank smart contracts will vary depending on the specific use cases that they are designed to support. However, there are some general functional requirements that are common to all central bank smart contracts. These requirements include:

- Authentication and authorization: Central bank smart contracts must be able to authenticate and authorize users before they can access and use the smart contract.
- Transaction processing: Central bank smart contracts must be able to process transactions efficiently and securely.
- Data management: Central bank smart contracts must be able to store and manage data securely and efficiently.
- Auditing and reporting: Central bank smart contracts must be able to generate auditable trails and reports to support transparency and accountability.

- Issue and manage currency: Central bank smart contracts must be able to issue new currency and to manage the supply of currency in circulation. This includes requirements for setting monetary policy, managing reserves, and conducting open market operations.
- Process payments: Central bank smart contracts must be able to process payments between financial institutions and individuals. This includes requirements for verifying the authenticity of payment instructions, ensuring that the payer has sufficient funds, and settling payments in real time.
- Implement monetary policy: Central bank smart contracts must be able to implement monetary policy tools, such as interest rate changes and quantitative easing. This includes requirements for setting and adjusting interest rates, purchasing and selling assets, and conducting other monetary policy operations.
- Provide financial services: Central bank smart contracts can be used to provide new financial services, such as central bank digital currencies (CBDCs) and programmable money. This would require requirements for managing the supply of CBDCs, facilitating payments between users, and programming money to perform specific tasks.

4.2 Non-Functional Requirements

Non-functional requirements (NFRs) for central bank smart contracts are the requirements that specify how a smart contract should perform, rather than what it should do. NFRs are typically divided into several categories, including performance, security, reliability, scalability, maintainability, and usability.

- Throughput: The smart contract must be able to process a high volume of transactions per second without becoming congested.
- Latency: The smart contract must be able to execute transactions quickly, with minimal latency.
- Resource utilization: The smart contract must be able to operate efficiently without using excessive resources, such as CPU, memory, and storage

- Confidentiality: The smart contract must protect the confidentiality of sensitive data, such as user identities and transaction amounts.
- Integrity: The smart contract must protect the integrity of data and transactions, preventing unauthorized modifications.
- Availability: The smart contract must be available 24/7, with minimal downtime.
- Auditability: The smart contract must be auditable, so that its code and execution can be reviewed for security vulnerabilities.

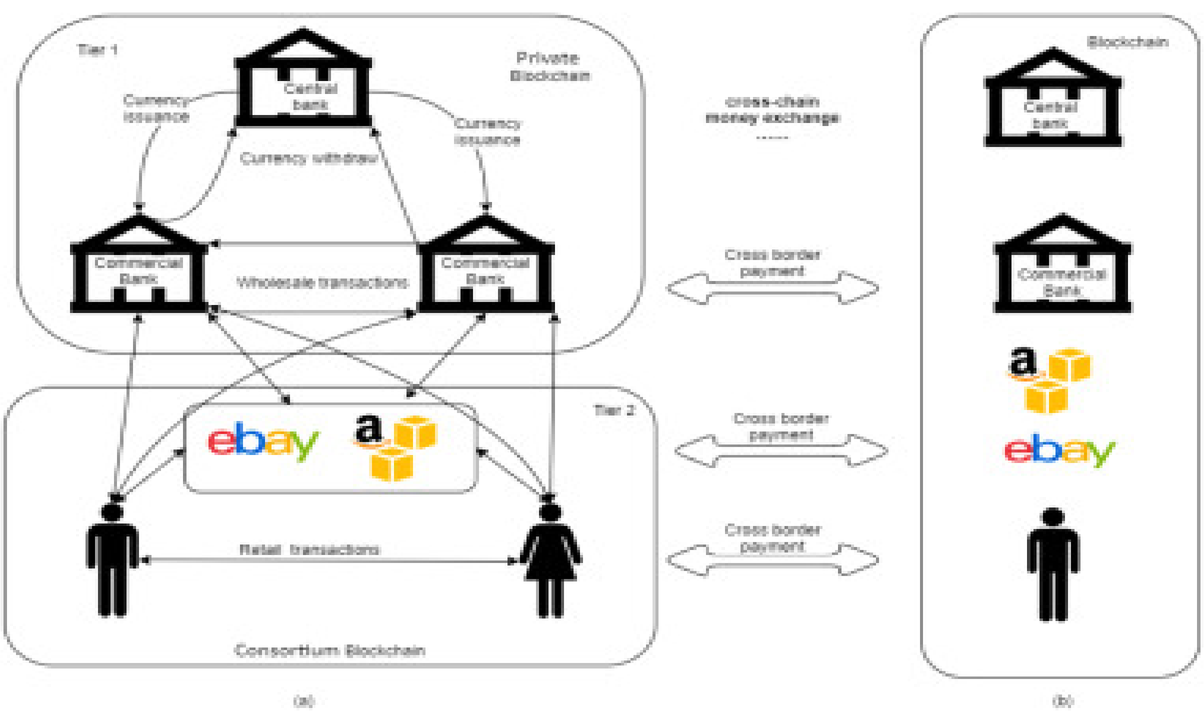
- Accuracy: The smart contract must produce accurate results, even in the event of errors or unexpected events.
- Consistency: The smart contract must produce consistent results, regardless of the order in which transactions are processed or the state of the blockchain.
- Robustness: The smart contract must be able to handle unexpected errors and events without crashing or losing data.

Scalability requirements for central bank smart contracts include:

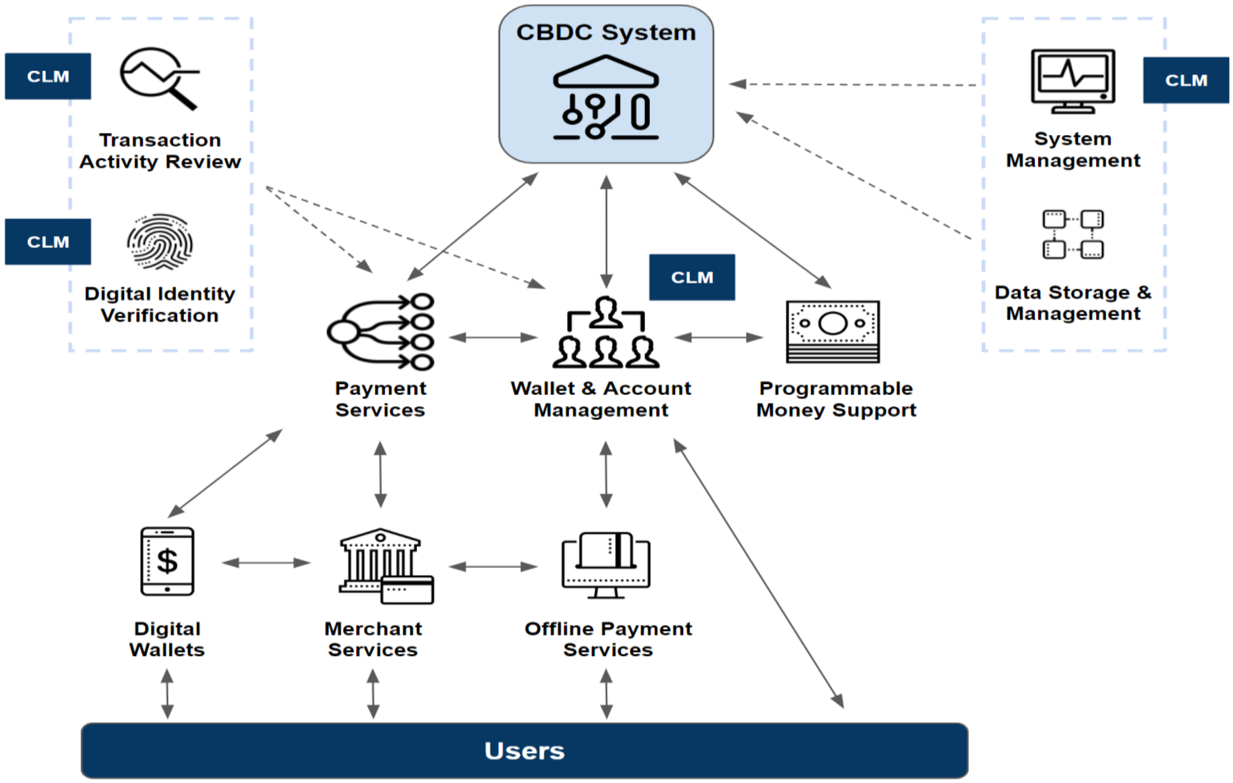
- Horizontal scalability: The smart contract must be able to scale horizontally by adding more nodes to the blockchain network.
- Vertical scalability: The smart contract must be able to scale vertically by improving the performance of individual nodes.
- Modularity: The smart contract should be modular, with well-defined interfaces between modules. This will make it easier to add new features and fix bugs.
- Documentation: The smart contract should be well-documented, so that developers can understand how it works and how to use it.
- Testability: The smart contract should be easy to test, so that developers can verify that it is working correctly.
- Simplicity: The smart contract should be easy to use, even for users who are not familiar with blockchain technology.
- Clearness: The smart contract should have a clear and concise user interface.
- Feedback: The smart contract should provide feedback to users about the status of their transactions and any errors that occur.

5. Project Design

5.1 Data Flow Diagram & User Stories

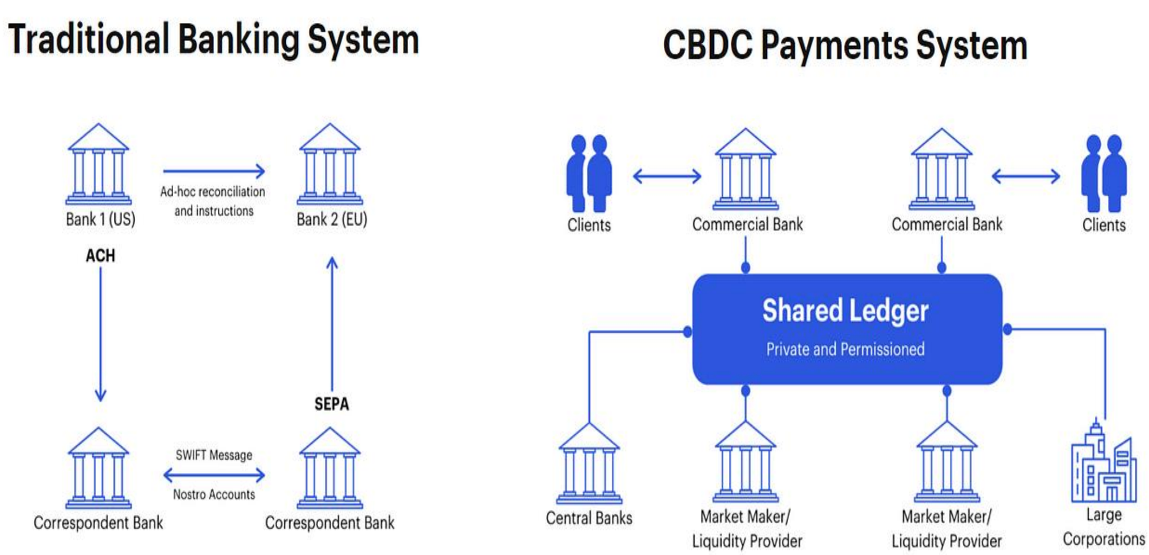


5.2 Solution Architecture



6. Project Planning & Scheduling

6.1 Technical Architecture



6.2 Sprint Planning & Estimation

Sprint planning and estimation for central bank smart contracts is a challenging task. Central bank smart contracts are complex systems that need to be highly secure, performant, and scalable. They also need to be compliant with a variety of regulations.

Sprint planning

1. Identify the sprint goals: The sprint goals should be specific, measurable, achievable, relevant, and time-bound. For example, a sprint goal for a central bank smart contract could be to implement a new feature that allows users to make payments using their CBDC.
2. Break down the sprint goals into tasks: The sprint goals should be broken down into smaller, more manageable tasks. Each task should be estimated in terms of the time and effort required to complete it.
3. Assign tasks to team members: Tasks should be assigned to team members based on their skills and experience.
4. Prioritize the tasks: The tasks should be prioritized based on their importance and urgency.
5. Create a sprint backlog: The sprint backlog is a list of all the tasks that need to be completed during the sprint. The sprint backlog should be updated throughout the sprint as needed.

Estimation

Estimation for central bank smart contracts is challenging because it is difficult to predict how long it will take to develop and test complex systems. However, there are a number of techniques that can be used to estimate the time and effort required for each task.

One common technique is to use expert judgment. This involves asking experienced developers to estimate the time and effort required for each task. Another technique is to use historical data. This involves looking at the time and effort required for similar tasks in the past.

A third technique is to use decomposition. This involves breaking down each task into smaller, more manageable subtasks. The time and effort required for each subtask can then be estimated using one of the other techniques.

Once the time and effort required for each task has been estimated, the total time and effort required for the sprint can be calculated. This information can then be used to set realistic expectations for the sprint and to make sure that the sprint goals are achievable.

Here are some specific tips for sprint planning and estimation for central bank smart contracts:

- Start by identifying the specific use cases that the central bank smart contract will support. This will help you to define the scope of the sprint and to identify the key features that need to be implemented.
- Break down the sprint goals into small, well-defined tasks. This will make it easier to estimate the time and effort required for each task and to track progress throughout the sprint.
- Use a variety of techniques to estimate the time and effort required for each task. This will help you to get a more accurate estimate and to identify any potential risks.

- Be realistic about the time and effort required to complete the sprint. Central bank smart contracts are complex systems, so it is important to allow enough time for development and testing.
- Communicate regularly with the team and with stakeholders throughout the sprint. This will help to ensure that everyone is on the same page and that any potential problems are identified and addressed early on.

6.3 Sprint Delivery Schedule

A sprint delivery schedule for central bank smart contracts will vary depending on the specific use cases that the smart contract will support and the complexity of the system. However, here is a sample sprint delivery schedule for a central bank smart contract that implements a new CBDC:

Sprint 1

Goal: Implement the core functionality of the CBDC, such as issuance, redemption, and transfers.

Tasks:

- Design the CBDC system architecture.
- Develop the smart contract code.
- Test the smart contract code.
- Deploy the smart contract to the blockchain network.

Sprint 2

Goal: Implement additional features and functionality for the CBDC, such as support for programmable money and conditional payments.

Tasks:

- Develop the smart contract code for the new features and functionality.
- Test the smart contract code.
- Deploy the updated smart contract to the blockchain network.

Sprint 3

Goal: Conduct integration testing with other financial systems and infrastructure.

Tasks:

- Integrate the CBDC smart contract with other financial systems and infrastructure.
- Test the integration.

Sprint 4

Goal: Conduct user acceptance testing and prepare for launch.

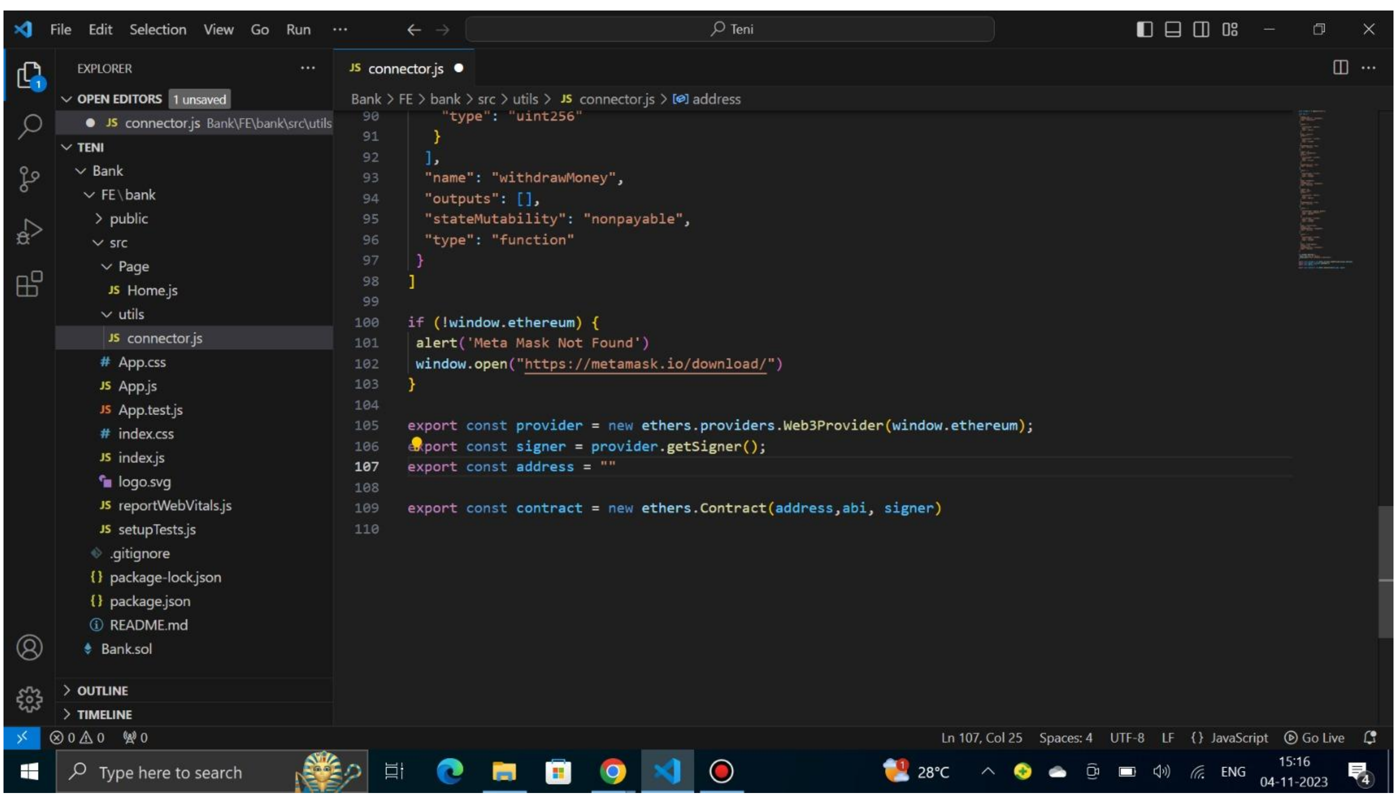
Tasks:

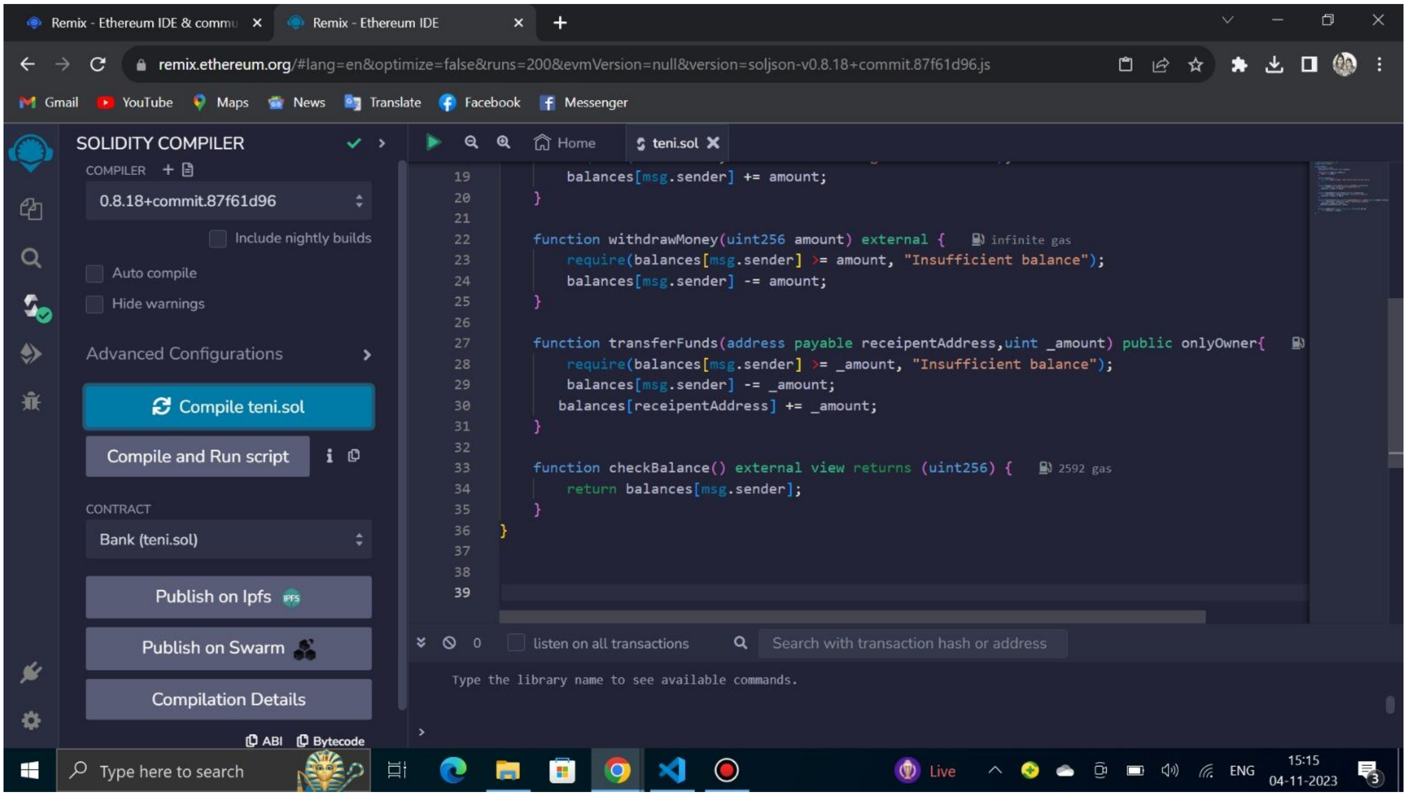
- Conduct user acceptance testing with a small group of users.
- Fix any bugs that are found.
- Prepare for the launch of the CBDC.

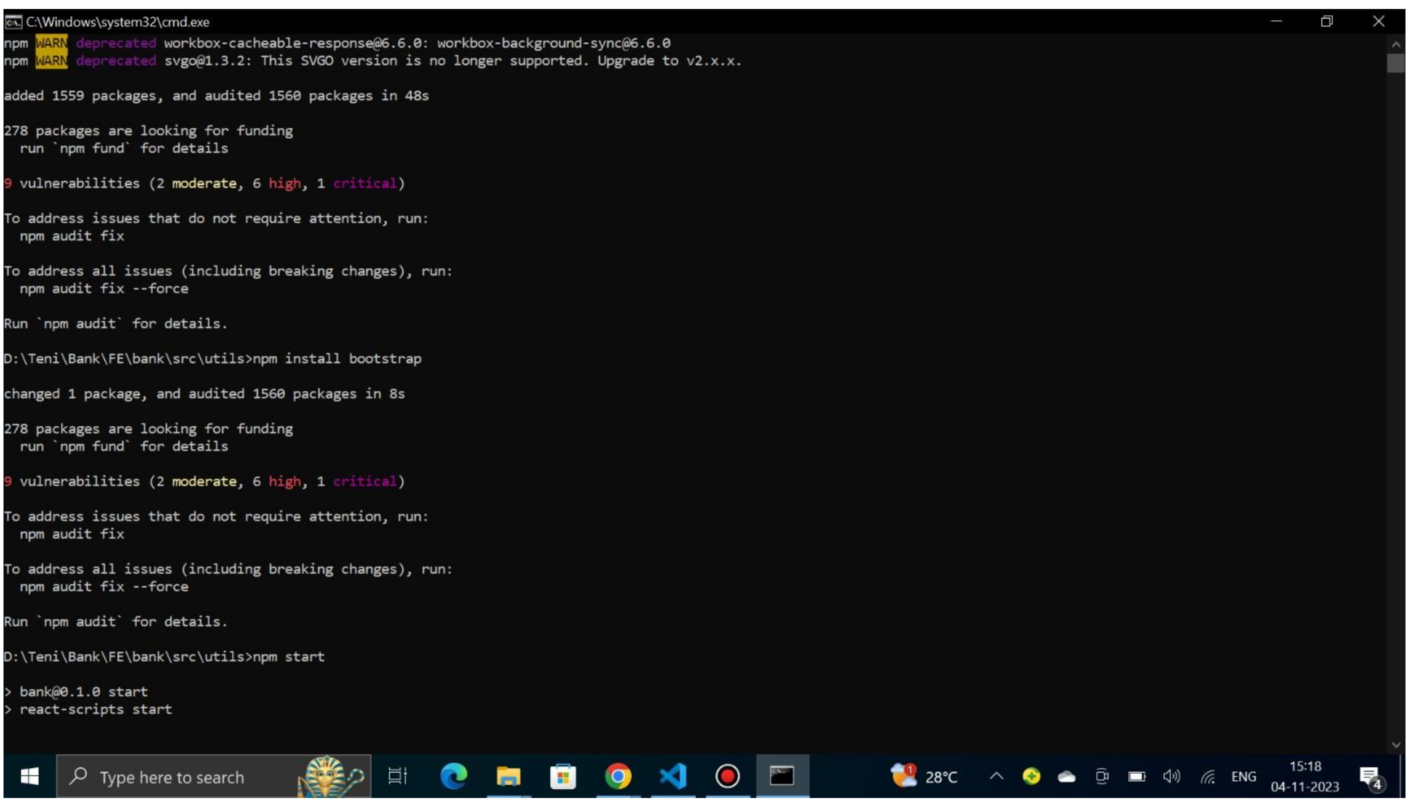
7. Coding & Solution

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

pragma solidity ^0.8.0;


contract Bank {

    address public owner;

    mapping(address => uint256) public balances;


    constructor() {

        owner = msg.sender;

    }


    modifier onlyOwner() {

        require(msg.sender == owner, "Only contract owner can call this");

        _;

    }


    function mintMoney(uint256 amount) external onlyOwner {

        require(amount > 0, "Amount must be greater than 0");

        balances[msg.sender] += amount;

    }


    function withdrawMoney(uint256 amount) external {
```

```
        require(balances[msg.sender] >= amount, "Insufficient balance");
        balances[msg.sender] -= amount;
    }

    function transferFunds(address payable receipentAddress,uint _amount) public onlyOwner{
        require(balances[msg.sender] >= _amount, "Insufficient balance");
        balances[msg.sender] -= _amount;
       balances[receipentAddress] += _amount;
    }

    function checkBalance() external view returns (uint256) {
        return balances[msg.sender];
    }
}
```




Remix - Ethereum IDE & commu
Remix - Ethereum IDE
remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.18+commit.87f61d96.js
Gmail YouTube Maps News Translate Facebook Messenger
SOLIDITY COMPILER
COMPILER
0.8.18+commit.87f61d96
Include nightly builds
Auto compile
Hide warnings
Advanced Configurations
Compile teni.sol
Compile and Run script
CONTRACT
Bank (teni.sol)
Publish on Ipfs
Publish on Swarm
Compilation Details
ABI
Bytecode
Home
teni.sol
balances[msg.sender] += amount;
}
function withdrawMoney(uint256 amount) external {
infinite gas
require(balances[msg.sender] >= amount, "Insufficient balance");
balances[msg.sender] -= amount;
}
function transferFunds(address payable receipentAddress,uint _amount) public onlyOwner{
require(balances[msg.sender] >= _amount, "Insufficient balance");
balances[msg.sender] -= _amount;
balances[receipentAddress] += _amount;
}
function checkBalance() external view returns (uint256) {
2592 gas
return balances[msg.sender];
}
}
listen on all transactions
Search with transaction hash or address
Type the library name to see available commands.
Type here to search
Live
ENG
15:15
04-11-2023


C:\Windows\system32\cmd.exe
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 1559 packages, and audited 1560 packages in 48s
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:\Teni\Bank\FE\bank\src\utils>npm install bootstrap
changed 1 package, and audited 1560 packages in 8s
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:\Teni\Bank\FE\bank\src\utils>npm start
> bank@0.1.0 start
> react-scripts start
Type here to search
28°C
ENG
15:18
04-11-2023

8. Performance Testing

Performance testing is a critical part of the software development process for central bank smart contracts. Central bank smart contracts need to be able to handle a high volume of transactions without becoming congested or expensive to use. They also need to be able to execute transactions quickly, with minimal latency.

There are a number of different performance testing techniques that can be used for central bank smart contracts. One common technique is to use a load testing tool to simulate a high volume of users accessing the smart contract. This can help to identify any bottlenecks in the system and to see how the smart contract performs under load.

Another common technique is to use a stress testing tool to simulate a large number of concurrent users accessing the smart contract. This can help to identify any single points of failure in the system and to see how the smart contract performs under stress.

In addition to these general performance testing techniques, there are also a number of specific performance tests that can be conducted for central bank smart contracts. For example, the following tests can be conducted:

- Transaction throughput testing: This test measures the number of transactions that the smart contract can process per second.
- Transaction latency testing: This test measures the time it takes for the smart contract to execute a transaction.
- Resource utilization testing: This test measures the amount of CPU, memory, and storage resources that the smart contract uses.
- Scalability testing: This test measures how the smart contract performs as the number of users and transactions increases.
- Security testing: This test identifies and addresses any security vulnerabilities in the smart contract code.

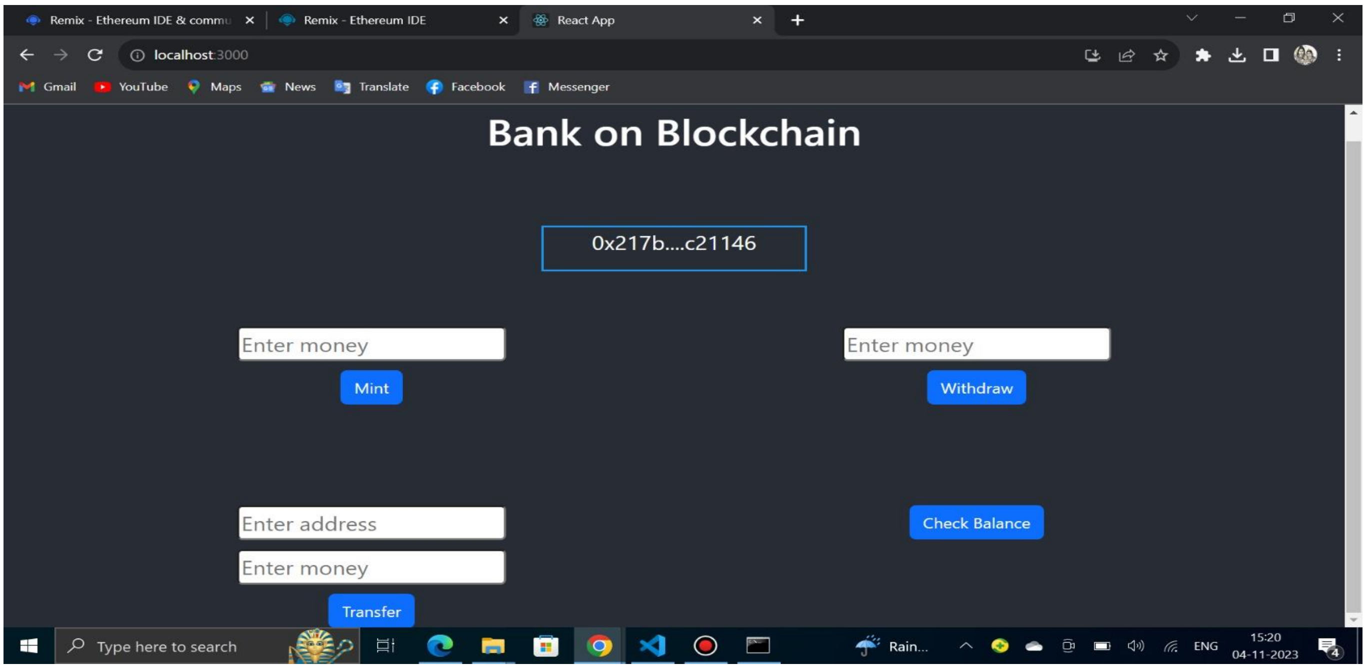
It is important to note that performance testing is not a one-time event. It should be conducted throughout the software development process, from the early stages of design to the final stages of deployment. This will help to ensure that the smart contract is able to meet its performance requirements.

Here are some specific tips for performance testing central bank smart contracts:

- Use a variety of performance testing techniques. This will help you to get a more comprehensive assessment of the smart contract's performance.
- Test the smart contract under realistic conditions. This includes using realistic workloads and simulating real-world user behavior.
- Use a variety of tools and frameworks to test the smart contract. This will help you to identify a wider range of potential performance problems.

- Document the results of your performance tests. This will help you to track the performance of the smart contract over time and to identify any areas where improvement is needed.

9. Result



10. Advantages & Disadvantages

Advantages

- Increased efficiency: Central bank smart contracts can automate and streamline central bank operations, such as currency issuance, payments processing, and monetary policy implementation. This can make central banks more efficient and reduce the risk of human error.
- Enhanced transparency: Central bank smart contracts can be made transparent and auditable, which can help to promote public trust in central banks.
- Improved financial inclusion: Central bank smart contracts can be used to create new financial products and services that are more accessible and affordable for people who are currently unbanked or underbanked.
- Enhanced financial stability: Central bank smart contracts can be used to implement new monetary policy tools and to reduce the risk of financial crises.
- New and innovative financial services: Central bank smart contracts can be used to create new and innovative financial services, such as programmable money and central bank digital currencies (CBDCs).

Disadvantages

- Complexity: Central bank smart contracts are complex systems that can be difficult to develop, test, and deploy.
- Security risks: Central bank smart contracts could be vulnerable to hacking and other security attacks.
- Privacy concerns: Central bank smart contracts could be used to track and monitor financial transactions, which could raise privacy concerns.
- Interoperability challenges: Central bank smart contracts need to be interoperable with existing financial systems and infrastructure.
- Regulatory uncertainty: There is still some uncertainty about the regulation of central bank smart contracts.

11. Conclusion

Central bank smart contracts are a new and emerging technology with the potential to revolutionize the way central banks operate and to provide new and innovative financial services to the public.

Central bank smart contracts can be used to automate and streamline central bank operations, enhance transparency, improve financial inclusion, enhance financial stability, and enable new and innovative financial services.

However, there are also a number of challenges and risks associated with central bank smart contracts, such as complexity, security risks, privacy concerns, interoperability challenges, and regulatory uncertainty.

Central banks are carefully considering the advantages and disadvantages of central bank smart contracts before adopting them on a wide scale. They are working to address the challenges and risks, such as developing security measures and ensuring interoperability. As the technology matures and the regulatory landscape becomes more clear, we can expect to see central banks adopt central bank smart contracts on a wider scale.

12. Future Scope

The future scope of central bank smart contracts is very promising. As the technology matures and the regulatory landscape becomes more clear, we can expect to see central banks adopt central bank smart contracts on a wider scale.

Here are some specific areas where central bank smart contracts are likely to have a significant impact in the future:

- Currency issuance and management: Central bank smart contracts can be used to automate and streamline the issuance and management of currency. This can make central banks more efficient and reduce the risk of human error.

- Payments processing: Central bank smart contracts can be used to process payments more quickly, securely, and efficiently. This can benefit both businesses and consumers.
- Monetary policy implementation: Central bank smart contracts can be used to implement monetary policy tools more precisely and efficiently. This can help central banks to achieve their macroeconomic goals.
- Financial inclusion: Central bank smart contracts can be used to create new financial products and services that are more accessible and affordable for people who are currently unbanked or under banked. This can help to promote financial inclusion and economic growth.
- Financial stability: Central bank smart contracts can be used to develop new tools for monitoring and managing financial risks. This can help to reduce the risk of financial crises.
- New and innovative financial services: Central bank smart contracts can be used to create new and innovative financial services, such as programmable money and central bank digital currencies (CBDCs). These new services can offer a number of benefits to businesses and consumers, such as increased efficiency, reduced costs, and new investment opportunities.

In addition to these specific areas, central bank smart contracts are also likely to have a broader impact on the financial industry as a whole. Central bank smart contracts have the potential to make the financial system more efficient, transparent, and accountable. They can also help to promote financial inclusion and enhance financial stability.

Overall, the future scope of central bank smart contracts is very promising. As the technology matures and the regulatory landscape becomes more clear, we can expect to see central banks adopt central bank smart contracts on a wider scale. This will lead to a number of benefits for businesses, consumers, and the economy as a whole.

13. Appendix

Github Link: https://github.com/Tenijio/Central-Bank-Smart-Contract.git

Demo Link : https://youtu.be/EFLTWE0i-8c?si=TxgngR0CqPoVx5MT