CENTRAL BANK SMART CONTRACT

A Project report submitted in partial fulfillment of 7th semester indegree of

BACHELOR OF ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING

Team ID: NM2023TMID03925

Submitted by

S.Karuppasamy	950320104015
S.Aron	950320104003
R. Jerrick Roshan	950320104018
S. Muthuramakrishnan	950320104303
M. Janaki selvam	950320104701

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GRACE COLLEGE OF ENGINEERING, THOOTHUKUDI

ANNA UNIVERSITY: CHENNAI 600025

1. INTRODUCTION

1.1 Project Overview:

Central Bank Smart Contract

The Central Bank Smart Contract initiative seeks to integrate the foundational role of Central Banks with the transformative capabilities of blockchain technology. Leveraging the security, transparency, and efficiency of decentralized ledgers, this project aims to enhance monetary operations, paving the way for potential Central Bank Digital Currencies (CBDC) issuance and streamlined policy implementations. By embedding features like transaction validation, monetary adjustments, and compliance verifications directly into smart contracts, we aim to bolster the agility and responsiveness of Central Bank operations, all while maintaining rigorous security and compliance standards. As this integration unfolds, challenges in technology and regulations will be addressed, ensuring that the fusion of traditional banking with blockchain remains seamless and future-ready.

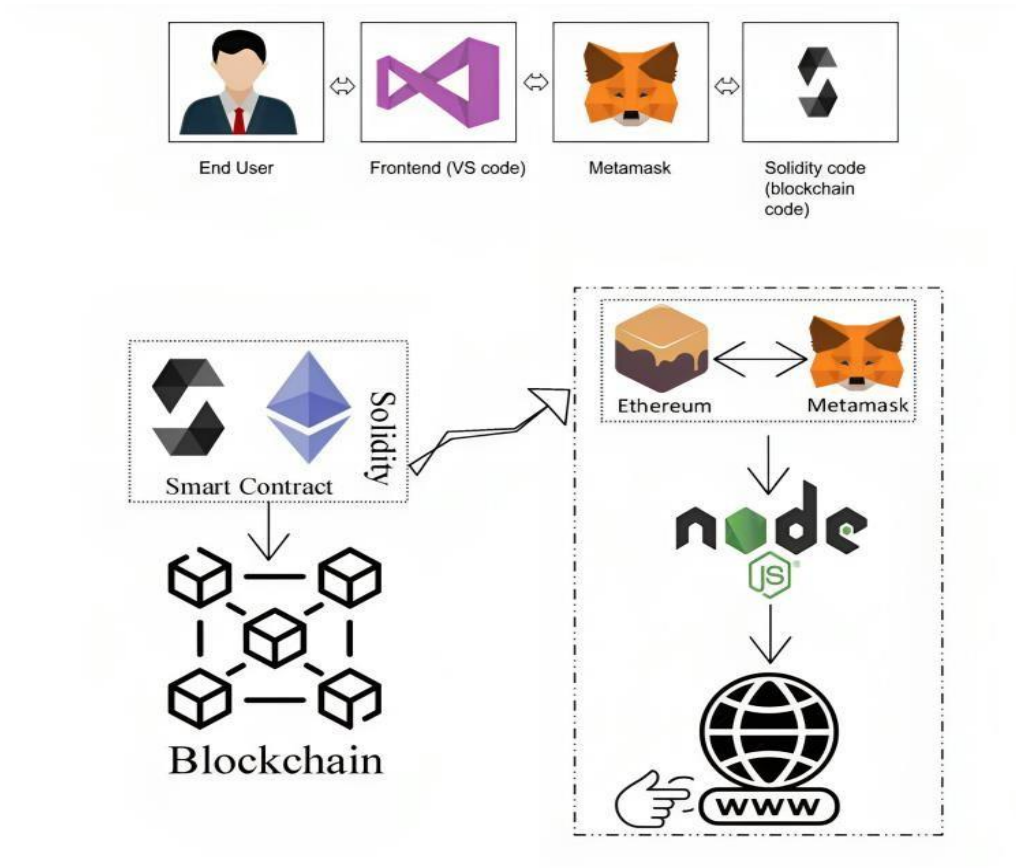


Fig: Solution Architecture Diagram

1.2 Purpose:

A blockchain-based electronic voting system can offer several advantages and purposes when compared to traditional paper-based or electronic voting systems. Here are some of the key purposes and benefits:

1. **Enhanced Monetary Policy Implementation:**

Smart contracts can automate the execution of monetary policy tools, such as open market operations, allowing for precise and timely policy adjustments to stabilize the economy.

2. **Improved Transparency:**

By recording central bank actions and transactions on a public blockchain, you can enhance transparency, enabling stakeholders and the public to monitor central bank activities in real time.

3. **Secure Payment System:** Implementing smart contracts in the payment system can enhance security and reduce fraud, making financial transactions more reliable and secure for businesses and individuals.

4. **Interoperability:** Ensure the smart contract can interact with other financial systems, both centralized and decentralized.

5. **Regulatory Compliance:** Ensure all financial transactions and operations align with the central bank's regulatory requirements and standards.

6. **Disaster Recovery and Resilience:** Using decentralized ledger technology can enhance the central bank's ability to recover and operate in case of system failures or disasters, ensuring financial system resilience.

7. **Stakeholder Engagement:** The project can engage stakeholders, such as financial institutions, government agencies, and the public, to ensure that the benefits of the central bank's smart contract initiatives are widely understood and appreciated.

8. **Regulatory Sandbox:**

Consider the establishment of a regulatory sandbox to allow innovative fintech companies to collaborate with the central bank in developing and testing smart contract solutions, promoting innovation in the financial sector.

2. LITERATURE SURVEY

2.1 Existing problem

Legacy Systems and Integration Challenges:

Explore literature discussing how the integration of smart contracts into legacy central bank systems can pose technical and operational challenges.

Scalability Issues:

Investigate studies on the scalability limitations of blockchain technology and how it may hinder the adoption of smart contracts by central banks.

Security Concerns:

Analyze research highlighting potential security vulnerabilities, including the risk of smart contract exploits and hacks, which central banks need to address.

Privacy and Confidentiality:

Examine publications that discuss privacy concerns related to blockchain technology and the potential exposure of sensitive financial data.

Regulatory and Legal Ambiguities:

Identify literature that explores the regulatory uncertainties and legal complexities associated with central bank smart contracts, including issues of jurisdiction.

Interoperability Challenges:

Review research on the lack of interoperability between different blockchain platforms and how this can hinder cross-border transactions and financial system integration.

Lack of Standardization:

Explore studies on the absence of standardized protocols and practices in the development of smart contracts for central banks.

Consensus Mechanisms and Energy Consumption:

Investigate how the choice of consensus mechanisms in blockchain networks can impact the energy consumption of central bank smart contract systems.

Financial Inclusion Barriers:

Identify the literature that discusses the challenges in ensuring that smart contracts and CBDCs promote financial inclusion for marginalized populations.

Cross-Border Regulatory Compliance:

Explore research on the complexities of complying with international regulations and anti-money laundering (AML) laws when using smart contracts for cross-border payments.

2.2 Reference

"Designing a Distributed Ledger Technology System for Interbank Settlements" published in the Review of Financial Studies.

Explores the application of DLT (Distributed Ledger Technology) for interbank settlements. Highlights the advantages of DLT over traditional systems. Discusses challenges in scalability and interoperability. Deep dives into the technical aspects of blockchain in finance.

Analysis of emerging trends like DeFi (Decentralized Finance).Studies on security and privacy concerns related to blockchain in banking.

"Mastering Blockchain" by Imran Bashir: A comprehensive guide covering the potential of blockchain in various industries, including banking. "Blockchain Basics: A Non-Technical Introduction in 25 Steps" by Daniel Drescher: Great for understanding the foundational

White Papers:

"The Bitcoin Whitepaper" by Satoshi Nakamoto: The foundational document for decentralized digital currencies.

Ethereum White Paper: Provides insights into smart contracts and decentralized applications.

Research Papers & Articles:

"Smart Contracts on the Blockchain – A Bibliometric Analysis and Review" published in the "Journal of King Saud University - Computer and Information Sciences."

"Banking on Blockchain: Costs Savings Thanks to the Blockchain Technology" - a study on how blockchain can influence cost savings in the banking sector.

2.3 Problem Statement Definition

Traditional financial systems often require intermediaries, leading to delays and increased transaction costs.

Current digital currency models may lack standardized regulations, governance, and verification processes. Potential risks associated with fraudulent activities, double-spending, and unauthorized creation of funds in the absence of a secure mechanism.

Need for a Solution:

To provide a decentralized yet regulated mechanism that ensures the authenticity and integrity of transactions related to central bank digital currencies (CBDCs).

To allow for programmable conditions to be associated with currency issuance, distribution, and transaction verifications, enhancing transparency and traceability.

Goals of the Central Bank Smart Contract:

Development of a secure and tamper-proof smart contract system to facilitate and manage CBDC transactions.

Reduction in the dependency on intermediaries by automating and ensuring compliance checks through the smart contract.

Provision of a transparent mechanism for auditing and monitoring CBDC transactions.

Central Bank and regulatory bodies:

To oversee and validate the smart contract's implementation.

Financial institutions: As potential integrators or users of the CBDC platform.

General public and businesses: As end-users and beneficiaries of the CBDC facilitated by the smart contract.

Developers and cybersecurity experts: To ensure the secure and efficient operation of the smart contract.

Desired Outcome:

A robust, secure, and efficient smart contract system that seamlessly integrates with the central bank's digital currency operations.

A platform that fosters trust among users, reduces transaction costs, and enhances the speed of transactions.

A transparent and easily auditable mechanism for regulators and other stakeholders.

User Accessibility and Interface:

Brief on user interface (UI) design and the user experience (UX) strategies to make the smart contract system user-friendly.

Pilot Testing and Feedback Loop:

Overview of pilot testing phases, targeted user groups, and mechanisms for gathering and implementing feedback.

Scalability and Future Integration:
Addressing the system's ability to handle growth in users and transactions and potential integrations with other financial platforms.

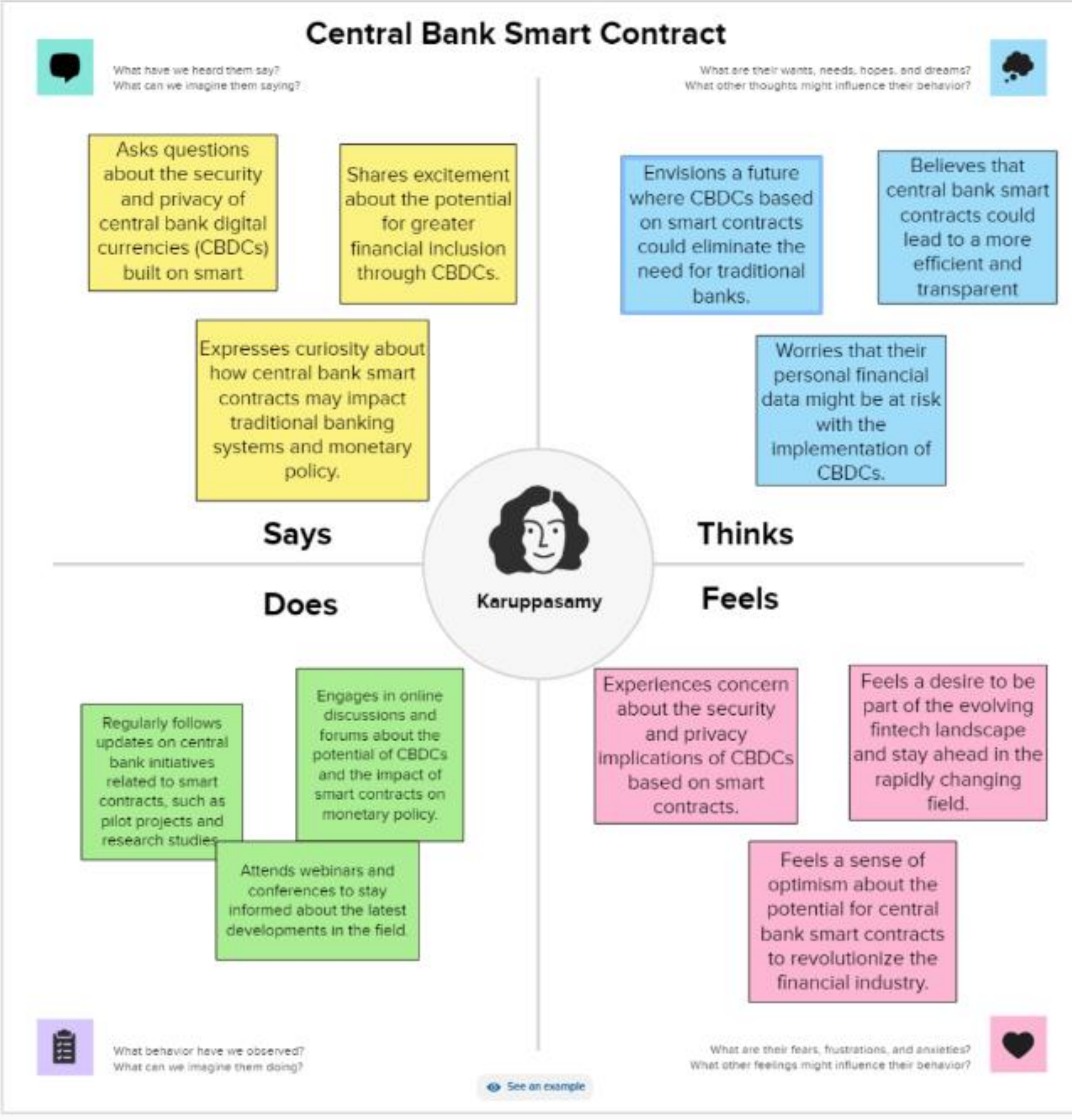
Training and Education:
Highlighting plans to educate stakeholders, financial institutions, and the public about the usage and benefits of the smart contract.

Risk Management and Mitigation:
Identifying potential risks related to the smart contract's operation and strategies to minimize or counteract those risks.

3. IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas

our primary stakeholders express a mix of excitement and apprehension. They recognize the potential benefits of enhanced transparency, security, and operational efficiency that smart contracts can bring to central banking, yet they also harbor concerns about security, privacy, and regulatory uncertainties. Stakeholders hope to achieve gains such as streamlined cross-border transactions and reduced operational costs, but they are wary of scalability limitations and the security vulnerabilities that blockchain technology can introduce. They value clear communication channels and expect the central bank to be responsive to their feedback. Key influencers in this space, including regulatory authorities and thought leaders, play a significant role in shaping stakeholder perspectives. The empathy map provides valuable insights into the emotional and practical considerations of our stakeholders, guiding our project's efforts to meet their expectations while addressing their concerns.

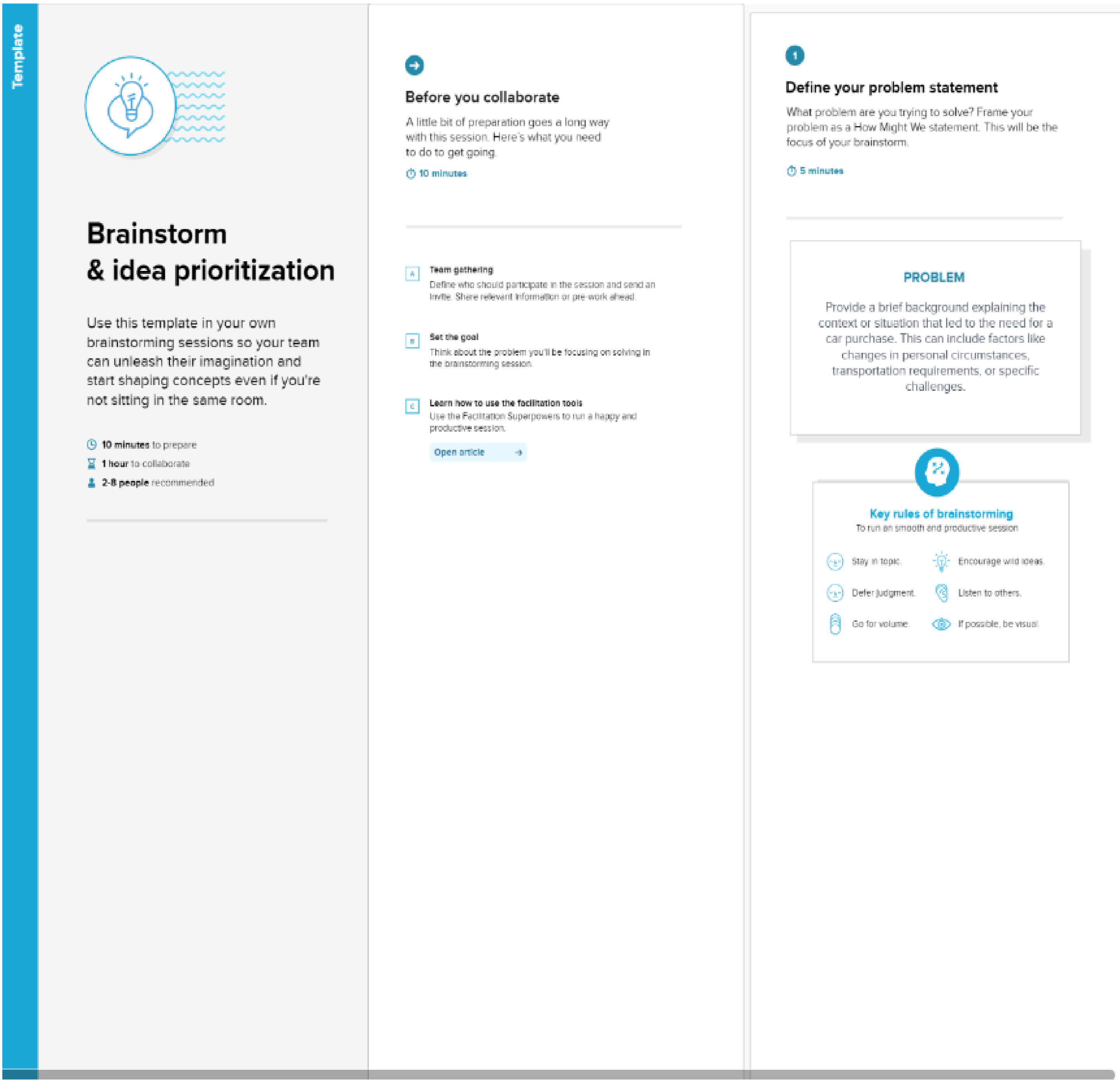


3.2 Ideation& Brainstorming:

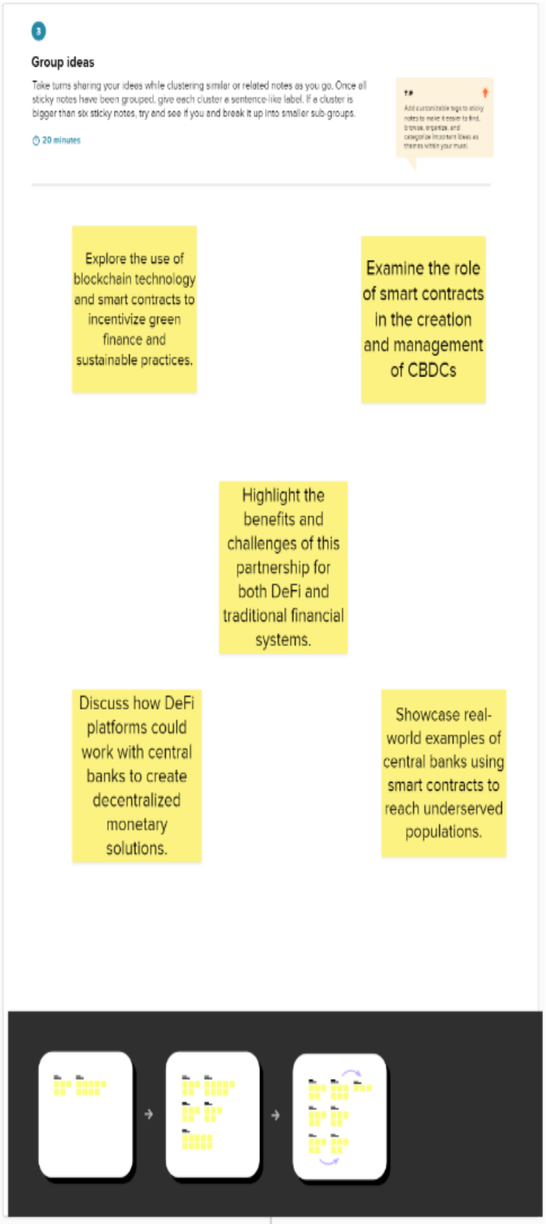
Brainstorm & Idea Prioritization Template:

In our quest to implement smart contracts within the central banking system to elevate efficiency, transparency, and security, our brainstorming and idea prioritization session was instrumental. We embarked on an open and creative brainstorming session, inviting participants to share their innovative ideas and suggestions. As a result, a multitude of ideas emerged, each holding great promise for the future of central banking. These ranged from leveraging blockchain technology to enhance monetary policy implementation to addressing challenges such as scalability and regulatory compliance. Our next step involves prioritizing these ideas by conducting a voting process, wherein participants allocate points based on importance and feasibility. The ideas with the highest points will guide our project teams as they develop action plans and set timelines for implementation. The brainstorming and prioritization process is an ongoing endeavor, fostering a culture of continuous improvement as we work towards modernizing our central bank's operations for the benefit of our economy and society.

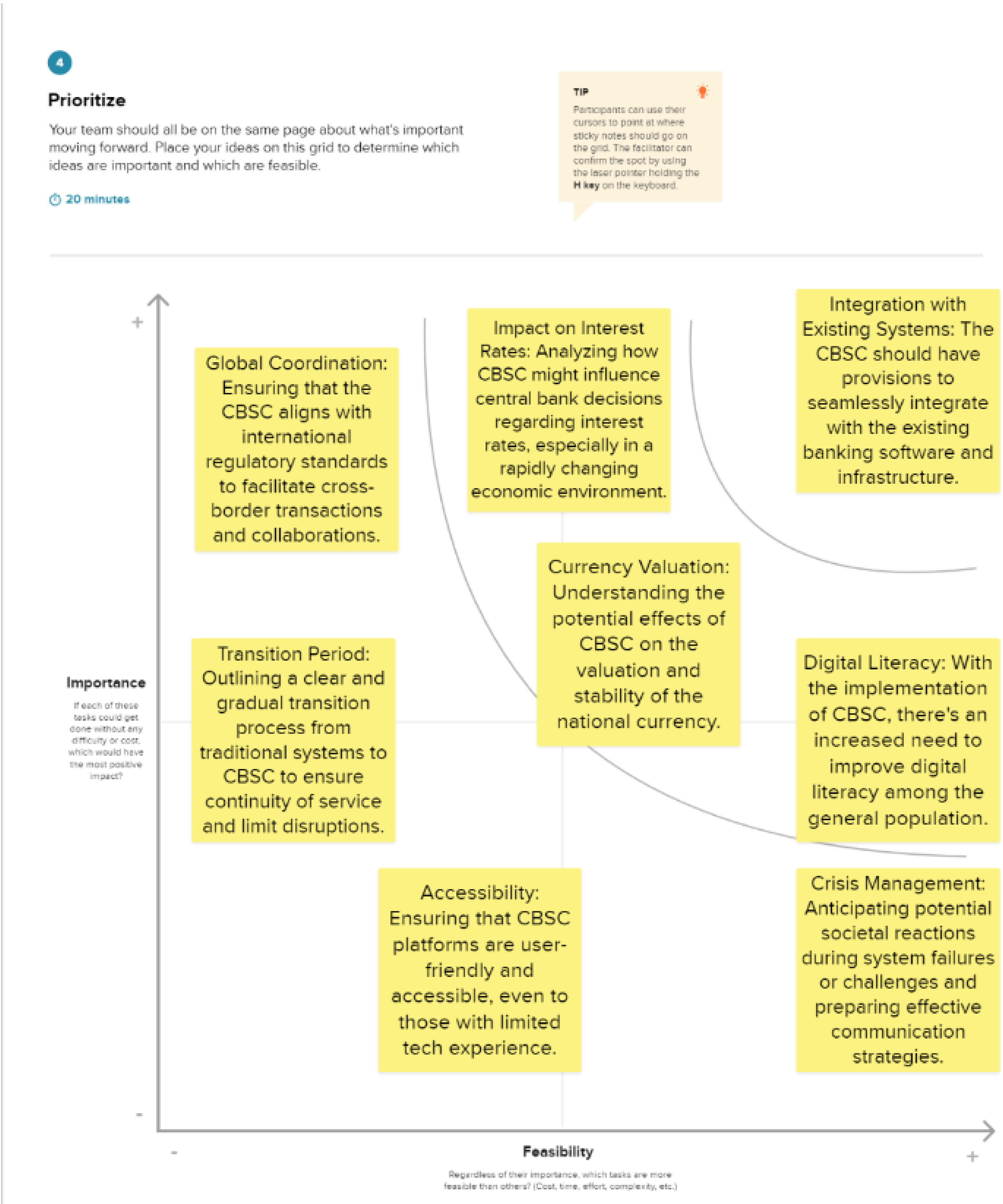
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization



4. REQUIREMENT ANALYSES

4.1 Functional Requirements:

Following are the functional requirements of the proposed solution

Requirement ID	Requirement Description	Priority	Notes
FR-001	The smart contract shall be able to mint and burn the central bank's digital currency (CBDC)	High	Essential for controlling the money supply.
FR-002	Allow authorized entities to create, manage, and revoke users' digital wallets	High	Only authorized entities should have this power.
FR-003	Enable real-time transaction validation and processing	High	Real-time operations are crucial for user confidence and functionality.
FR-004	Provide an interface for auditing and reporting transaction histories	High	For transparency and regulatory compliance.
FR-005	Enable cross-border transaction processing	Medium	To ensure CBDC can be used in international transactions.
FR-006	Integrate with existing bank systems to allow for transfer between CBDC and traditional bank	Medium	Seamless integration is key for adoption.
FR-007	Offer multi-signature transaction validation for high-value transactions	Medium	To enhance security for significant transfers.

4.2 Non-functional Requirements:

Following are the non-functional requirements of the proposed solution

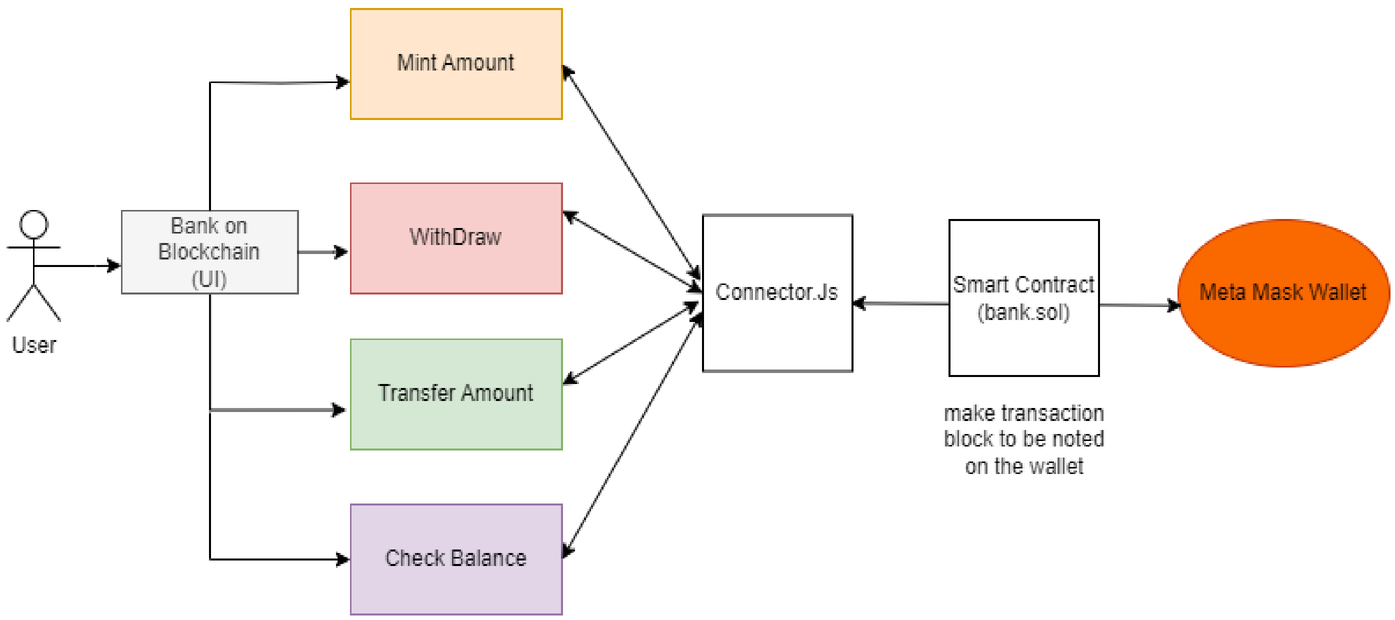
Requirement ID	Requirement Description	Priority	Notes
NFR-001	The smart contract shall execute transactions within 3 seconds of initiation	High	Speed is essential for user experience.
NFR-002	The system shall maintain 99.99% uptime	High	High availability ensures trust and reliability.
NFR-003	Data stored shall be encrypted using state-of-the-art encryption algorithms	High	To ensure security and privacy of transactions and balances.
NFR-004	The smart contract code shall undergo regular third-party security audits	High	Regular security assessments ensure vulnerabilities are identified and addressed.
NFR-005	The user interface shall be intuitive and user-friendly	Medium	Ease of use will encourage adoption and regular use by both institutions and individuals.
NFR-006	The system shall provide timely notifications for user actions and system updates	Medium	Notifications ensure users are informed and can take timely actions

5. PROJECT DESIGN:

5.1 Data Flow Diagram

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

Example: DFD Level 0 (Industry Standard)



5.2 Solution Architecture

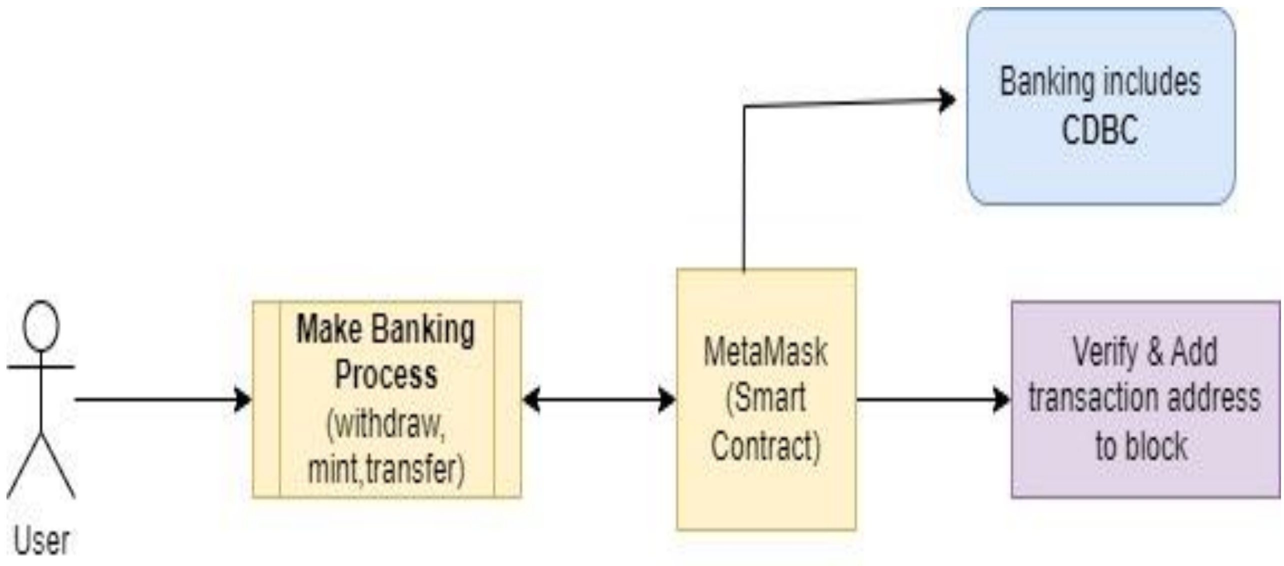
Solution Architecture:

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, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.

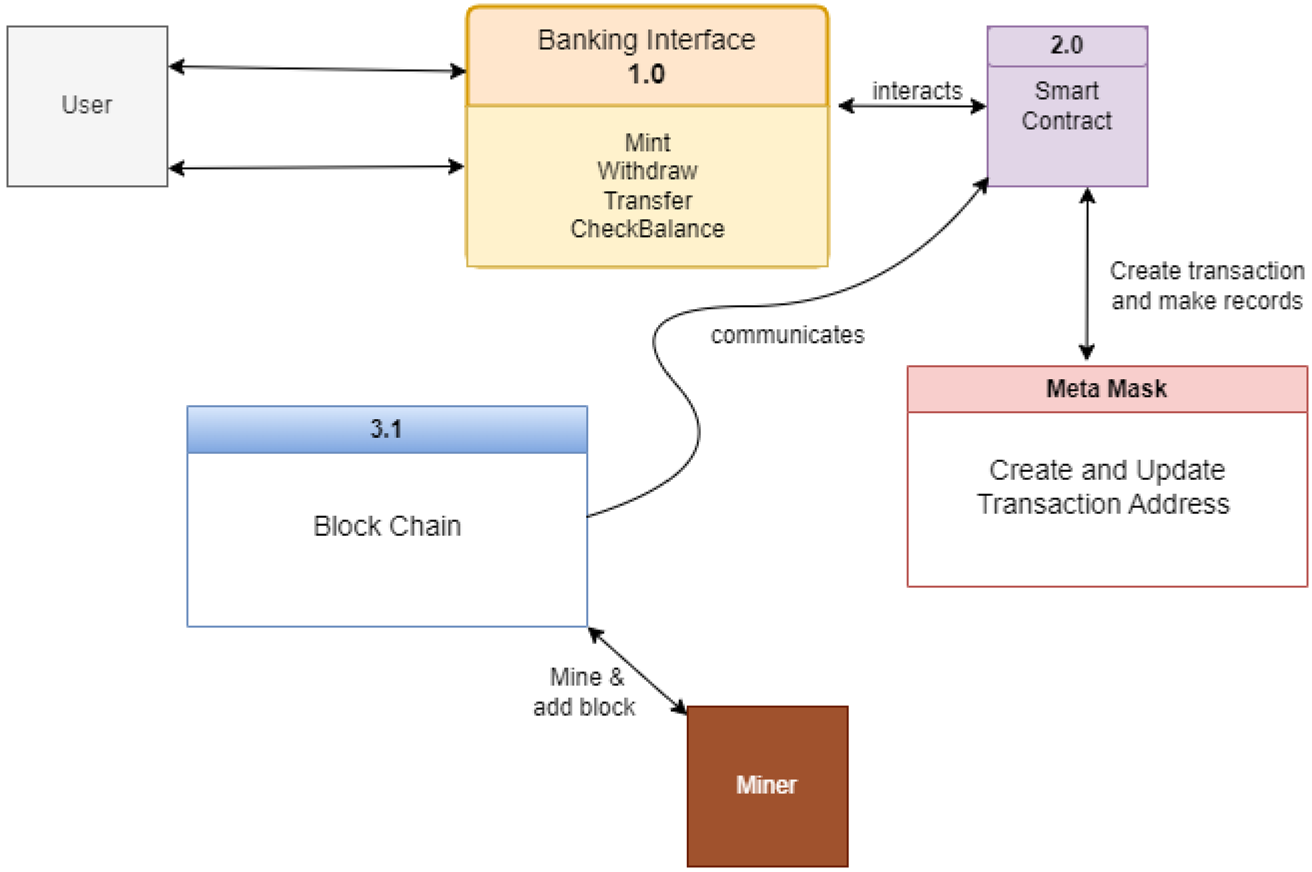
Example: Solution Architecture Diagram:



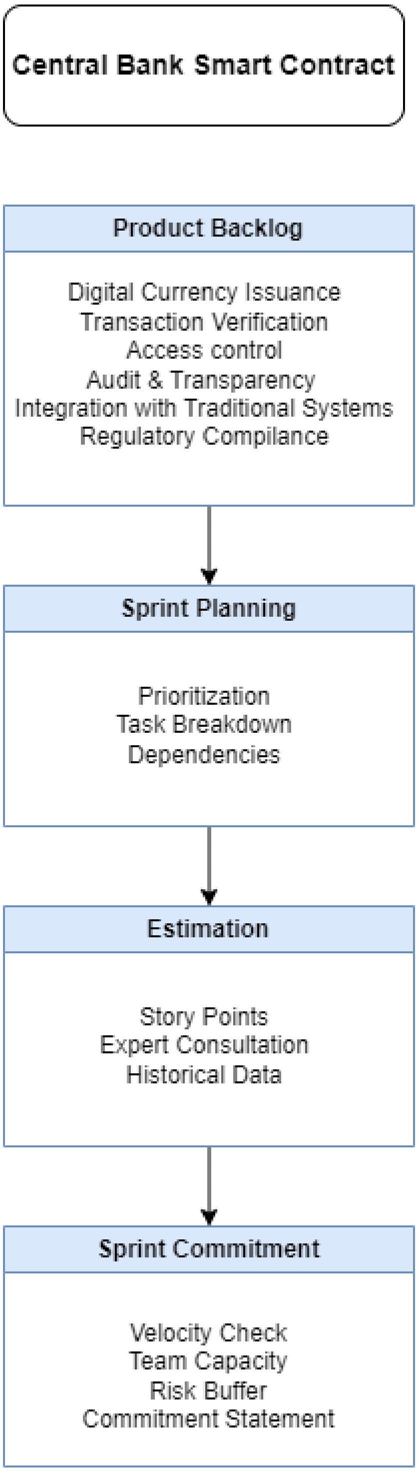
6 . PROJECTPLANNING AND SCHEDULING:

6.1 Technical Architecture:

Technical Architecture (TA) is a form of IT architecture that is used to design computer systems. It involves the development of a technical blueprint with regard to the arrangement, interaction, and interdependence of all elements so that system-relevant requirements are met.

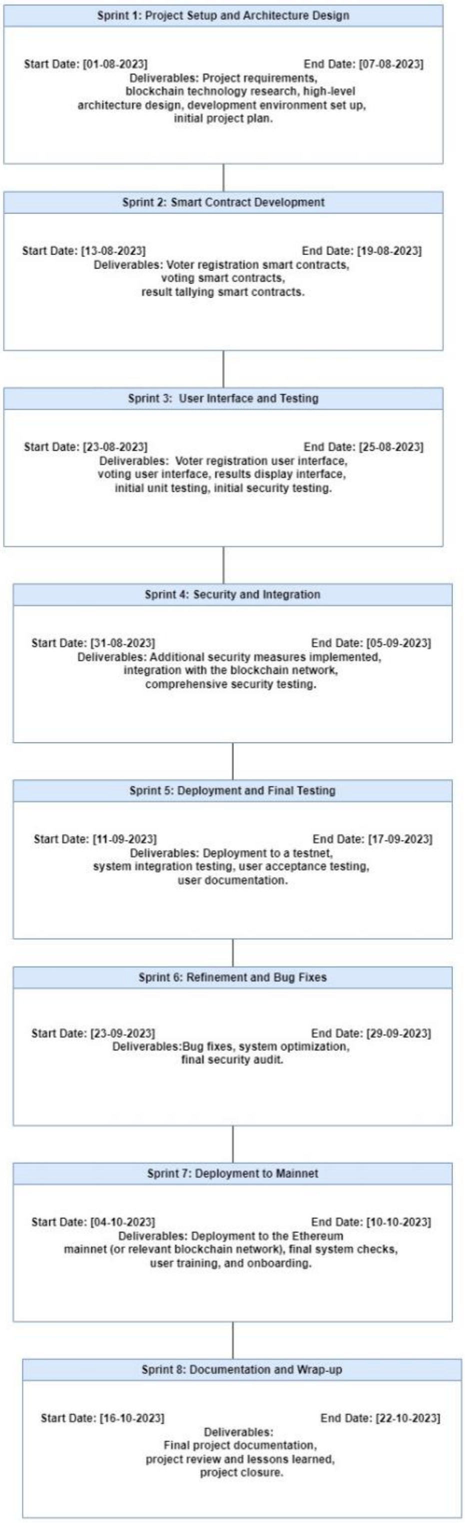


6.2 Sprint Planning & Estimation



Central Bank Smart Contract
Product Backlog
Digital Currency Issuance
Transaction Verification
Access control
Audit & Transparency
Integration with Traditional Systems
Regulatory Compilance
Sprint Planning
Prioritization
Task Breakdown
Dependencies
Estimation
Story Points
Expert Consultation
Historical Data
Sprint Commitment
Velocity Check
Team Capacity
Risk Buffer
Commitment Statement

6.3 Spring Delivery Schedule



7. CODING & SOLUTIONS:

7.1 Connector with wallet connect:

Connector.js

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

const abi = [
 {
  "inputs": [],
  "stateMutability": "nonpayable",
  "type": "constructor"
 },
 {
  "inputs": [
   {
    "internalType": "address",
    "name": "",
    "type": "address"
   }
  ],
  "name": "balances",
  "outputs": [
   {
    "internalType": "uint256",
    "name": "",
    "type": "uint256"
   }
  ],
  "stateMutability": "view",
  "type": "function"
 },
 {
  "inputs": [],
  "name": "checkBalance",
  "outputs": [
   {
    "internalType": "uint256",
    "name": "",
    "type": "uint256"
   }
  ],
```

```
 "stateMutability": "view",
 "type": "function"
},
{
 "inputs": [
  {
   "internalType": "uint256",
   "name": "amount",
   "type": "uint256"
  }
 ],
 "name": "mintMoney",
 "outputs": [],
 "stateMutability": "nonpayable",
 "type": "function"
},
{
 "inputs": [],
 "name": "owner",
 "outputs": [
  {
   "internalType": "address",
   "name": "",
   "type": "address"
  }
 ],
 "stateMutability": "view",
 "type": "function"
},
{
 "inputs": [
  {
   "internalType": "address payable",
   "name": "receipentAddress",
   "type": "address"
  },
  {
   "internalType": "uint256",
   "name": "_amount",
```

```
    "type": "uint256"
   }
  ],
  "name": "transferFunds",
  "outputs": [],
  "stateMutability": "nonpayable",
  "type": "function"
 },
 {
  "inputs": [
   {
    "internalType": "uint256",
    "name": "amount",
    "type": "uint256"
   }
  ],
  "name": "withdrawMoney",
  "outputs": [],
  "stateMutability": "nonpayable",
  "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 = "0x35bEc26BF70f653752331e21f2a21b781b956F62"

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

7.2 Banking Process (Mint, Transaction,Withdraw,CheckBalance)

```
 import React,{useState} from "react";
import {Button,Container,Row,Col} from 'react-bootstrap';
import 'bootstrap/dist/css/bootstrap.min.css';
import { contract } from "../utils/connector";
```

```
function Home() {
const [Money, setMoney] = useState("");
const [withdrawMoney, setWithdrawMoney] = useState("");
const [transferMoney, setTransferMoney] = useState("");
const [Bal, setBal] = useState("");
const [address, setaddress] = useState("");
const [Wallet, setWallet] = useState("");


  const handleAddress = (e) => {
    setaddress(e.target.value)
  }

 const handleMoney = (e) => {
  setMoney(e.target.value)
 }

 const handleWithdrawMoney = (e) => {
  setWithdrawMoney(e.target.value)
 }

 const handleTransferMoney = (e) => {
  setTransferMoney(e.target.value)
 }

 const checkBalance = async () => {
   let bal = await contract.checkBalance()
    setBal(bal.toString())
 }

 const mint = async () => {
  try {
    let tx = await contract.mintMoney(Money.toString())
    let txWait =  await tx.wait()
    alert(txWait.transactionHash  )
  } catch (error) {
   alert(error)
  }
 }
```

```
const withdraw = async () => {
  try {
    let tx = await contract.withdrawMoney(withdrawMoney.toString())
    let txWait = await tx.wait()
    alert(txWait.transactionHash)
  } catch (error) {
    alert(error)
  }
}

const transfer = async () => {
  try {
    let tx = await contract.transferFunds(address,transferMoney.toString())
    let txWait = await tx.wait()
    alert(txWait.transactionHash)
  } catch (error) {
    alert(error)
  }
}

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"}}>Bank 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>
     <Row>
      <Col>
       <div>
        {/* <label>Mint Money</label><br /> */}
        <input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleMoney} type="number" placeholder="Enter money" value={Money} /> <br />
        <Button onClick={mint} style={{ marginTop: "10px" }} variant="primary">Mint</Button>
       </div>
      </Col>
      <Col>
       <div>
        {/* <label>Withdraw Money</label><br /> */}
        <input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleWithdrawMoney} type="number" placeholder="Enter money" value={withdrawMoney} /> <br />
        <Button onClick={withdraw} style={{ marginTop: "10px" }} variant="primary">Withdraw</Button>
       </div>
      </Col>
     </Row>
     <Row style={{marginTop:"100px"}}>
      <Col>
       <div>
        {/* <label>Address</label><br /> */}
        <input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleAddress} type="string" placeholder="Enter address" value={address} /> <br />

        {/* <label>Transfer Money</label><br /> */}
```

```
        <input style={{ marginTop: "10px", borderRadius: "5px" }}
onChange={handleTransferMoney} type="number" placeholder="Enter money"
value={transferMoney} /> <br />
        <Button onClick={transfer} style={{ marginTop: "10px" }}
variant="primary">Transfer</Button>
       </div>
      </Col>
      <Col>
       <div>
        {/* <label>Check Balance</label><br /> */}
        <Button onClick={checkBalance} style={{ marginTop: "10px" }}
variant="primary">Check Balance</Button>
        <p>{Bal}</p>
       </div>
      </Col>
      </Row>
    </Container>

  </div>
  )
}

export default Home;
```

Source Code:

https://drive.google.com/file/d/1wFf3E7U0So7ZedJMSmeEKzjMFRgxhriH/view?usp=sharing

8.PERFORMANCE TESTING:

8.1 Performance Metrics

1.Transaction Processing Speed:

Measurement of the time it takes to record and validate a vote on the blockchain. A faster processing speed ensures that the voting process is efficient and responsive, reducing voter wait times and enhancing user experience.

2.Scalability:

The system's ability to handle a large number of concurrent voters and transactions. Scalability is crucial to accommodate growing voter populations and to ensure that the system remains responsive under high loads during elections.

3.Security and Data Integrity:

The system's ability to protect against unauthorized access, tampering, or manipulation of voting data. Security metrics include the number of attempted breaches, the detection of anomalies, and the system's ability to maintain the integrity of the vote records.

4.Voter Accessibility and Usability:

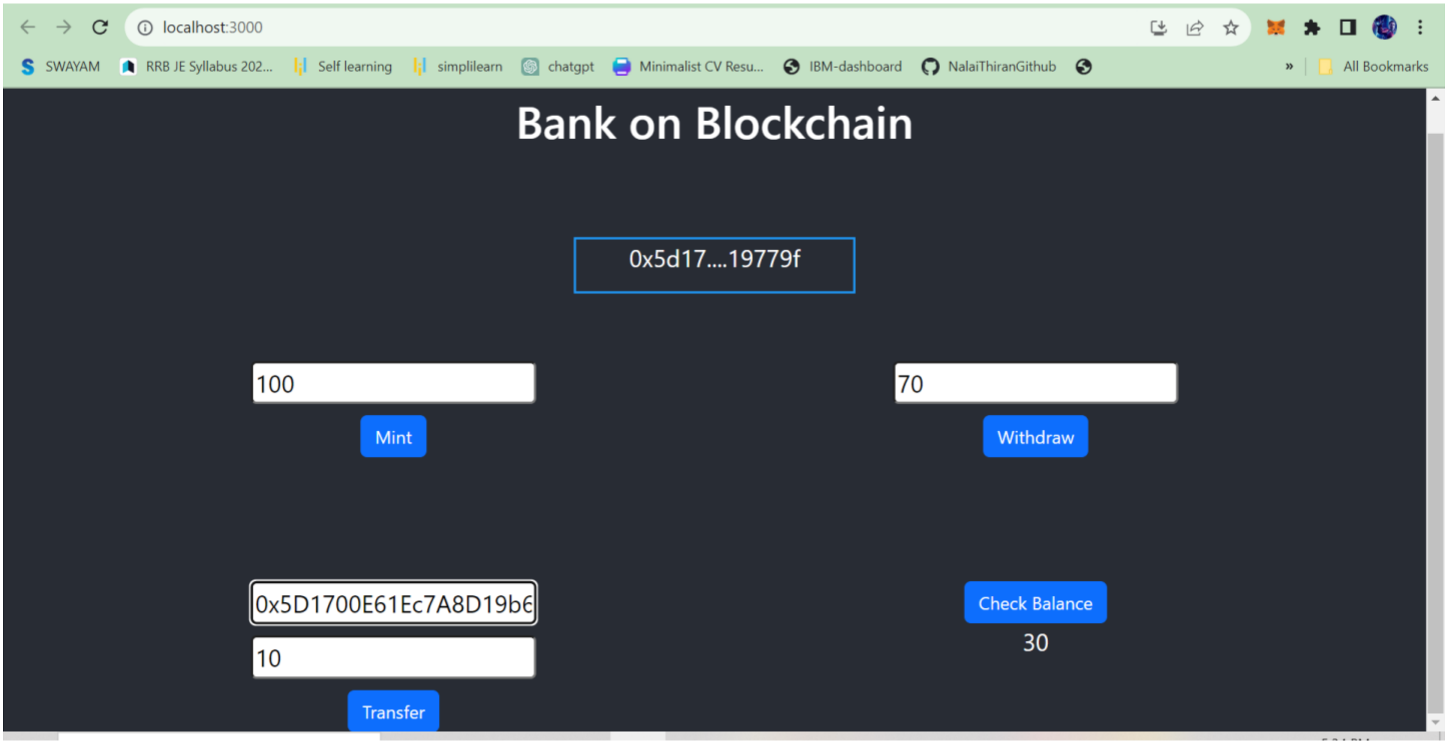
Evaluation of how accessible and user-friendly the system is for a diverse range of voters. Metrics could include the percentage of successfully registered voters, the average time taken to cast a vote, and feedback from voters regarding the user interface and overall experience.

5.Fault Tolerance and Redundancy:

Assessment of the system's ability to continue functioning in the event of hardware or network failures. Metrics can include system uptime during an election, the number of instances of system recovery, and the redundancy mechanisms in place to ensure uninterrupted service

9.Result:

9.1 Output Screenshots



10. ADVANTAGES AND DISADVANTAGES

10.1 Advantages of Central Bank Smart Contract

1.Reduced Operational Bottlenecks:

Streamlined Operations: With automated and predefined rules, many banking processes can be expedited, reducing delays.

Real-time Settlement: Transactions can be settled in real-time, eliminating the waiting period inherent in many traditional systems.

2. Enhanced Financial Inclusion

Accessible Services: As smart contracts can operate 24/7, they offer banking services round-the-clock, beneficial especially for populations in different time zones or with limited access to traditional banking.

Microtransactions: They can efficiently handle small transactions, enabling more people to engage in banking activities without prohibitive fees.

3. Dynamic Policy Adaptation

Automated Responses: Monetary policies can be programmed into the contract to automatically respond to certain economic triggers or indicators.

Timely Interventions: Allows for quicker reactions to economic shifts, potentially stabilizing economies faster than manual interventions.

4. Reduced Reliance on Intermediaries

Direct Transactions: Enables peer-to-peer transactions without the need for intermediaries, reducing costs and potential points of failure.

Increased Trust: As operations are automated and verifiable, it reduces the need to rely on third-party entities, fostering trust in the system.

5. Enhanced Data Integrity & Reporting

Immutable Records: Once a transaction is recorded on the blockchain, it cannot be altered, ensuring data integrity.

Automated Reporting: Smart contracts can generate real-time reports, aiding in decision-making and ensuring transparency.

Disadvantages of Central Bank Smart Contract:

1.Technical Complexity:

Implementing and maintaining smart contracts requires specialized expertise, which might not be readily available.

2. Irreversibility:

Once executed, smart contracts cannot be easily altered, which can be problematic if errors or unforeseen circumstances arise.

3. Scalability Concerns:

Handling a large number of transactions typical for central banks might raise scalability issues on certain blockchain platforms.

4. Regulatory Uncertainties:

The evolving nature of blockchain regulation might pose challenges for central bank implementations.

5. Integration with Existing Systems:

Integrating smart contracts with traditional banking systems can be challenging and may require significant changes to existing infrastructures.

11.CONCLUSION:

Central Bank Smart Contracts stand at the crossroads of finance and technology, heralding a new era in the realm of monetary operations and policy implementation. As financial systems globally strive for higher efficiency, transparency, and security, these smart contracts offer a promising pathway. Their capability to autonomously execute predefined rules can streamline many of the banking operations, from routine transactions to more intricate monetary policies.

However, with any nascent technology, there are bound to be teething issues. The immutable nature of smart contracts, while offering transparency, also presents challenges, particularly if flaws or loopholes are discovered after deployment. The integration of such a disruptive technology into established banking systems calls for meticulous planning, rigorous testing, and an adaptive regulatory environment.

Furthermore, the broader public's understanding and acceptance of such a system play a pivotal role in its success. For central banks, this means not only ensuring the robustness of the technology but also educating stakeholders and ensuring adequate safeguards.

In conclusion, while Central Bank Smart Contracts hold immense promise, their successful implementation will depend on a harmonious blend of technological expertise, regulatory foresight, and public trust. The journey towards full-scale adoption might be complex, but the potential rewards in reshaping the banking landscape make it a pursuit worth undertaking.

12. FUTURE SCOPE:

The future scope of blockchain-based banking systems is promising and holds the potential to address various challenges in the current banking processes. Here are some key aspects of its future scope:

Interbank Transactions:

Smart contracts can facilitate seamless interbank transactions, leading to faster and more efficient global banking operations.

Digital Central Bank Currencies (CBDCs)

Many countries are considering or already piloting CBDCs, where smart contracts will play a pivotal role.

Real-time Policy Implementation:

Smart contracts could allow central banks to implement monetary policy changes in real-time, reacting promptly to economic conditions.

Collaboration with FinTech:

As the FinTech sector grows, collaboration between central banks and these innovators could lead to novel solutions and applications of smart contracts.

Integration with IoT and AI:

In the future, the amalgamation of smart contracts with Internet of Things (IoT) devices and Artificial Intelligence (AI) could bring about unprecedented automation and intelligence in banking operations.

13. APPENDIX:

Source Code:

https://drive.google.com/drive/folders/1t8zycjB3zXQdCW4780o_1F1rx9My5juR
?usp=sharing

GitHub:

https://github.com/Karuppasamy13/Naanmudhalvan-Blockchain-
NM2023TMID03925.git

Demo Video Link:

https://drive.google.com/file/d/1-
plCpxzmL04LCOHUdpAndZ5oxM9zHYS6/view?usp=sharing