Kathford International College of Engineering and Management

Balkumari, Lalitpur, Nepal

A

Project Proposal Report

on

DeVote: A Decentralized Voting System Using Custom Blockchain

Submitted To

Department of Information Technology

Kathford International College of Engineering and Management

In partial fulfillment of the requirement for the Bachelor Degree in Computer Science and Information Technology

Submitted By

Satish Bajagain (28491/078)

Pritam Das (28485/078)

Shreeya Palikhel (28494/078)

May, 2025

TABLE OF CONTENTS

DeVote: Decentralized Voting System Using Custom Blockchain 1

1. INTRODUCTION 1

2. PROBLEM STATEMENT 1

3. OBJECTIVES 2

4. METHODOLOGY 2

a. Requirement Identification 2

i. Study of Existing Systems 2

ii. Requirement Analysis 5

b. Feasibility Study 9

i. Technical Feasibility 9

ii. Operational Feasibility 9

iii. Economic Feasibility 9

iv. Schedule (Gantt chart showing the project timeline) 10

c. High-Level System Design 10

ii. Flowchart of the Proposed System 13

iii. Description of algorithm 14

5. EXPECTED OUTCOME 18

6. REFERENCES 18

LIST OF FIGURES

Figure i: UseCase Diagram for DeVote 6
Figure ii: Project Timeline for Devote (May-August, 2025) 10
Figure iii: Architecture of the DeVote system 10
Figure iv: Illustration of working mechanism of DeVote in a flowchart 13

1. INTRODUCTION

Free and fair elections are essential to democracy, yet current paper and digital voting systems remain vulnerable to fraud, tampering, and a lack of transparency. Existing blockchain-based solutions, like those on Ethereum, struggle with high costs, scalability issues, and inadequate privacy, making them impractical for large-scale elections. DeVote addresses these challenges with a purpose-built decentralized voting system that ensures security, privacy, and scalability without relying on third-party platforms.

Unlike general-purpose blockchains, DeVote uses a lightweight, custom-designed architecture optimized for elections. It combines zero-knowledge proofs (zk-SNARKs) to validate votes without revealing identities, ring signatures for anonymous authentication, and Merkle trees to guarantee vote integrity. A consensus mechanism, PBFT, protects against attacks while maintaining fault tolerance [1].

By prioritizing tamper-proof records, full transparency, and strong voter privacy, DeVote offers a practical, secure solution for modern elections that restores trust in democracy.

2. PROBLEM STATEMENT

Existing voting system whether paper-based or electronic, suffer from critical vulnerabilities that undermine electoral integrity, security, and trust. These limitations originate from centralized control, inadequate security mechanisms, insufficient privacy guarantees, and scalability bottlenecks, making them unfit for modern democratic processes.

a. Centralization & Lack of Verifiability

Current systems rely on a single point of control, making them susceptible to manipulation, technical failures, and insider threats. Voters lack cryptographic proof that their ballots were recorded or tallied correctly, as centralized authorities manage vote storage and counting without transparent, auditable mechanisms.

b. Security Vulnerabilities

Centralized databases are prime targets for cyberattacks, including:

- Data tampering (altering/deleting votes)
- Denial-of-Service (DoS) attacks, disrupting election processes
- Insider threats, where malicious actors manipulate results without detection

c. Weak Voter Privacy

Many systems fail to cryptographically decouple voter identity from ballot content, risking exposure through:

- Inadequate encryption
- Linkable voter records
- Lack of zero-knowledge verification

d. Scalability Limitations

- Traditional systems struggle with high voter turnout, leading to delays and errors.
- Blockchain-based solutions (e.g., Ethereum) face:
    - High transaction costs (gas fees)
    - Low throughput (limited TPS)
    - Growing ledger storage, increasing node operational overhead

e. Inefficiencies in Public Blockchains

While blockchain provides immutability and decentralization, existing implementations are ill-suited for voting due to:

- Slow finality times, delaying result confirmation
- Consensus inefficiencies (e.g., PoW/PoS bottlenecks)
- Lack of specialized privacy-preserving mechanisms (e.g., zk-proofs, ring signatures)

3. OBJECTIVES

- To develop a secure, blockchain-based voting system ensuring vote immutability, voter anonymity (via zk-SNARKs/Ring Signatures), and end-to-end verifiability (using Merkle trees) while maintaining a user-friendly interface.
- To implement robust authentication (NID-based) and decentralized validation (PBFT) for tamper-proof, privacy-preserving, and efficient elections.

4. METHODOLOGY

a. Requirement Identification

i. Study of Existing Systems

This comprehensive review analyzes existing blockchain voting systems, from academic prototypes to deployed solutions, providing critical insights for decentralized voting system development. The analysis covers architectural approaches, consensus mechanisms, privacy solutions, and implementation experiences across public, private, and hybrid blockchain designs.

Blockchain Voting System Architectures

Follow My Vote (c.2016)

Follow My Vote represents an early blockchain voting initiative built on the BitShares blockchain-a delegated proof-of-stake platform forked from Bitcoin-with a "voting DAC" (Decentralized Autonomous Corporation) design.
The system uses blind signatures for ballot casting and pseudonymous keypairs for identity verification. Voters generate public/private key pairs, submit blinded tokens to a registrar for signing, then later unblind and cast vote tokens on-chain. A distinctive feature is the "forgiveness" mechanism, allowing vote changes before a deadline while maintaining one-person-one-vote integrity through the blockchain ledger.
Despite offering full decentralization with end-to-end verifiability and open-source transparency, Follow My Vote remained largely theoretical. Its main limitations include requiring a trusted registrar to verify eligibility (introducing potential centralization) and complexity in blind-signature shuffling and key management. The organization advocates for a fully online blockchain-based voting system where voters can use personal devices including computers and mobile phones [2].

BitCongress (c.2014)

BitCongress utilized multiple public blockchains-Bitcoin and Counterparty-for voting operations. The system issued "Vote Coins" as mined vote tokens, verified by miners in cryptocurrency 2.0 layers. Each vote was essentially a Bitcoin transaction with a fee incentive for miners.
The architecture combined several cryptographic technologies including "Bitcoin, BitMessage, BitTorrent, Proof Of Existence, Reddit & Ethereum". Within this framework, the Ethereum network tallied and verified smart contracts labeled as VoteCoins (YES, NO, & NEUTRAL votes) in a blockchain register, maintaining a public record of all votes.
The primary challenge with BitCongress was its minimal privacy protection-all votes were public ledger transactions-and the system effectively functioned as an economic democracy where vote-buying was technically possible, rather than ensuring one-person-one-vote principles [3].

Chirotonia E-Voting Framework

Chirotonia represents a more recent academic implementation, currently serving as the official e-voting platform at the University of Napoli Federico II in Italy. The system combines blockchain with linkable ring signatures based on Elliptic Curve Cryptography (ECC) to produce compact cryptographic keys and digital signatures.
The voting protocol divides the process into six distinct phases: organization, registration, opening, voting, closing, and counting. Each phase involves different actors: voters, organizers, identity managers, and confidentiality managers.
A notable security improvement in Chirotonia is its recent integration of distributed key generation to address the previous single point of failure. Before this enhancement, the committee generated an RSA key pair, publishing the public key for vote encryption and securely storing the private key for later decryption. The improved design incorporates ETHDKG (Ethereum Distributed Key Generation) protocol, distributing responsibility among multiple entities to enhance security and fault tolerance.

Privacy and Security Approaches Used Above

Open Vote Network with zk-SNARKs

The Open Vote Network demonstrates advanced privacy protection through zero-knowledge proofs. Implemented on Ethereum, this system uses zk-SNARKs to verify off-chain computations without revealing sensitive information.
The architecture includes three primary zk-SNARK arithmetic circuits: publicKeyGen, encryptedVoteGen, and Tallying-corresponding to key generation, vote encryption, and result tallying. The design performs computations over a cyclic group with a finite field on the Baby Jubjub elliptic curve, optimized for performance.
Registration requires voters to submit public keys with zero-knowledge proofs, allowing verification without exposing private information. The protocol then enables encrypted vote casting with mathematical guarantees of correctness.

Voatz Mobile Voting Platform

Voatz represents one of the few blockchain voting systems deployed in official U.S. elections. The application connects to a permissioned Hyperledger Fabric network running approximately 32 validator nodes distributed across AWS and Azure datacenters.
The system architecture emphasizes identity verification through a multi-layered approach:

1. Voters download the mobile app and register with government-issued ID
2. Face/fingerprint biometric authentication binds the voter to their device
3. AES256-encrypted votes are sent to the Fabric network
4. Each block contains anonymized vote records, ensuring immutability
5. Voters receive visual receipts for personal verification

While Voatz demonstrates strong identity controls and employs geographically distributed nodes for resilience, critics note its highly centralized governance-states ultimately control the blockchain, and the ledger lacks public auditability. Security researchers have also questioned potential vulnerabilities in the mobile app and device trust chain.

Luxoft e-Vote (Zug, Switzerland 2018)

The Luxoft e-Vote implementation for the City of Zug represents a notable government-sanctioned blockchain voting experiment. Built on Hyperledger Fabric, the system is uniquely integrated with Zug's existing Ethereum-based digital identity system (uPort) for voter authentication.
Architecturally, the system distributed Fabric nodes across three data centers (two in Switzerland, one in Ireland) for redundancy. Votes were anonymized through encryption techniques, likely homomorphic or mix-net style, allowing tallying without linking ballots to voters.
Though limited in scale (approximately 240 registered voters with 72 participants) and non-binding, the pilot demonstrated feasibility for local democratic processes and showed high user satisfaction in post-vote surveys.

Agora (Swiss, c.2018)

Agora employs a sophisticated multi-layer blockchain architecture:

1. A permissioned "Cothority" network collects votes
2. Data is anchored to Bitcoin ("Cotena" ledger) for immutability
3. A public "Bulletin Board" ledger enables end-to-end verification
4. A "Valeda" network of citizen auditors running IPFS validates results

The system uses threshold ElGamal encryption for ballots and Neff shuffling for anonymity. This design enables voters to encrypt ballots under a threshold key, which are then anonymously shuffled, posted to the blockchain bulletin board, and finally decrypted for tallying.
Agora was used in March 2018 as a third-party observer for Sierra Leone's presidential election, recording 280 polling locations' paper ballots on-chain as an independent audit mechanism.

ii. Requirement Analysis

Functional Requirements

- The system must authenticate voters using National ID verification.
- Use cryptographic techniques such as Ring Signatures to anonymize voter identity during login.
- Users must be able to cast a vote that is encrypted and signed.
- Enforce one vote per voter by preventing double voting through blockchain validation.
- Store all votes immutably on a custom-built blockchain.
- Validate each vote’s eligibility and correctness using zero-knowledge proofs (zk-SNARKs) without revealing voter identity.
- Implementation of PBFT, a consensus mechanism to confirm votes securely and efficiently.
- Organize votes in Merkle Trees to guarantee data integrity and enable efficient verification of recorded votes.
- Use homomorphic encryption to tally votes without decrypting individual ballots, preserving confidentiality.

USE CASE DIAGRAM

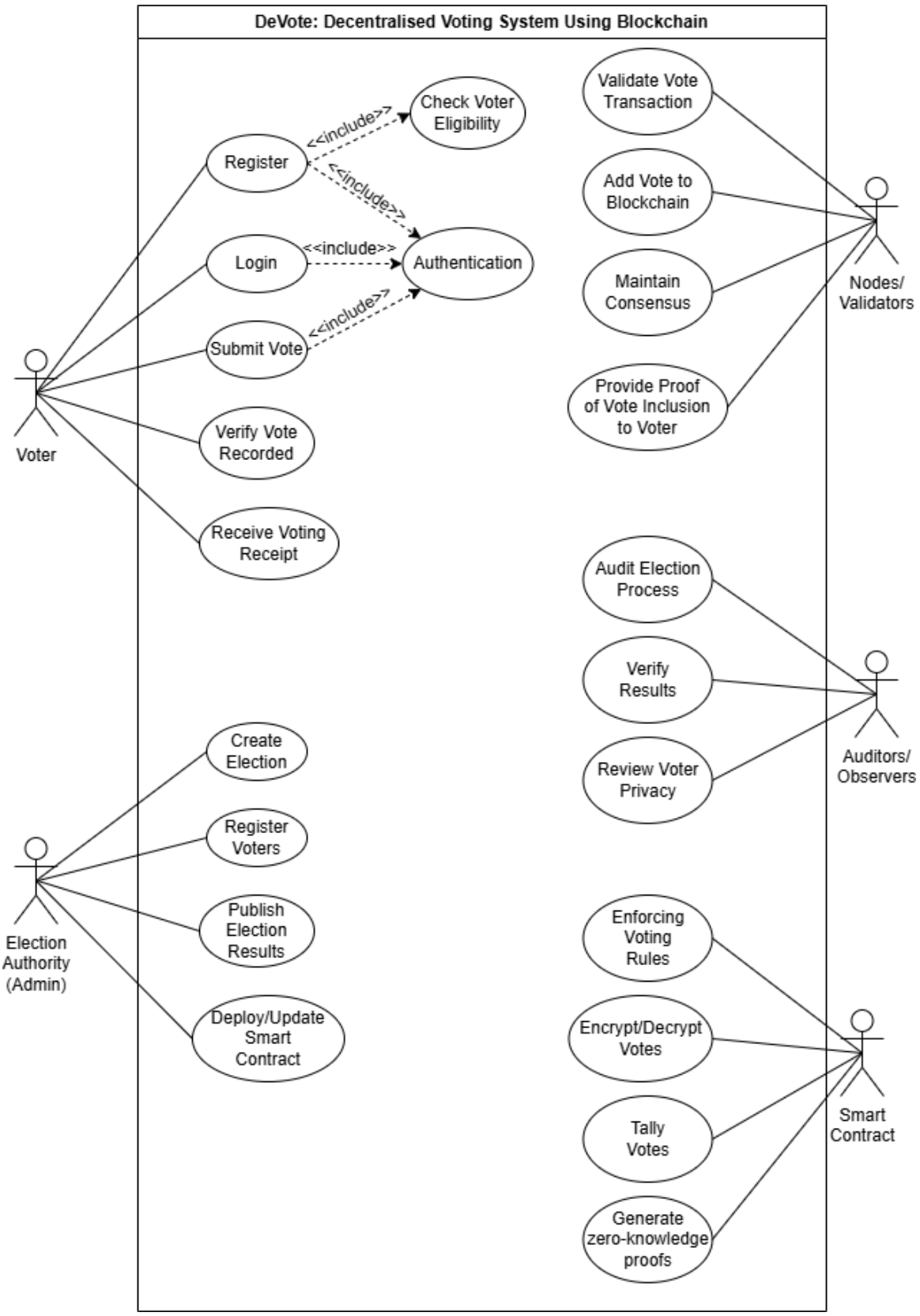

Figure i: UseCase Diagram for DeVote

1. Key Actors

1. Voter – The person who casts a vote.
2. Election Authority – The trusted entity responsible for election setup and management.
3. Blockchain Network (Nodes/Validators) – The distributed network validating and storing votes.
4. Auditor/Observer – External party verifying election integrity and transparency.
5. Smart Contract – The automated system enforcing election rules on the blockchain.

2. Core Use Cases

2.1 For Voter:

1. Register to Vote – Register identity, with NID verification.
2. Authenticate Voter – Prove eligibility without revealing identity (use zk-SNARK verification).
3. Cast Vote – Submit encrypted vote to the blockchain.
4. Verify Vote Cast – Confirm that their vote is recorded (cryptographic proof).

2.2 For Election Authority:

1. Create Election – Define election parameters: candidates, voting period, rules.
2. Register Voters – Approve and register voters to participate.
3. Publish Election Results – Publish tally securely after election closure.
4. Manage Smart Contract – Deploy and monitor election smart contract.

2.3 For Blockchain Network (Nodes/Validators):

1. Validate Vote Transaction – Verify vote authenticity and encryption correctness.
2. Add Vote to Blockchain – Append validated vote transaction to the chain.
3. Maintain Consensus – Run consensus protocol (PBFT) to secure the ledger.

4. Generate Proof of Vote Inclusion – Provide cryptographic proof that a vote is included.

2.4 For Auditor/Observer:

1. Audit Election Process – Check election integrity and transparency.
2. Verify Results – Confirm correctness of vote tally and blockchain data.
3. Review Voter Privacy – Ensure zero-knowledge proofs and privacy mechanisms are intact.

2.5 For Smart Contract:

1. Enforce Voting Rules – Ensure voters cannot vote multiple times.
2. Encrypt/Decrypt Votes – Handle cryptographic operations for vote privacy.
3. Tally Votes – Automatically count votes post-election.
4. Generate Zero-Knowledge Proofs – Provide proofs for vote correctness without revealing vote content.

Non-functional Requirements

- Ensure robust security using advanced cryptography to protect against tampering, fraud, and cyberattacks.
- Maintain strict privacy by concealing voter identities and choices with zk-SNARKs, Ring Signatures, and Bulletproofs.
- Guarantee that once a vote is cast, it cannot be altered or deleted (immutability).
- Design the scalable system to handle large-scale elections with many voters simultaneously.
- Provide an easy-to-use, accessible and responsive interface suitable for non-technical users.
- Enable integration with government NID databases and other relevant systems.
- Offer transparency through an auditable system that protects voter privacy while allowing public verification.
- Deliver consistent and reliable performance even under heavy network loads or partial failures.
- Comply with applicable election laws, privacy regulations, and standards to ensure legality and public trust.

b. Feasibility Study

i. Technical Feasibility

The technical feasibility of the DeVote system is strongly supported by the maturity and reliability of the underlying technologies. This project implements several well-known technologies like zk-SNARKs, homomorphic encryption, Merkle trees and Ring Signatures which are all prevalent for both privacy-focused blockchain and digital security platforms. These technologies provide the core foundations for voter anonymity, vote integrity, and data validation. Moreover, the project is based on a custom-built blockchain eliminating the dependency on external platforms like Ethereum that have high transaction costs and less scalability. Also, PBFT (Byzantine Fault Tolerance) ensures fault tolerance in distributed networks.

ii. Operational Feasibility

The DeVote system is operationally designed to be user-centric and compatible with existing systems for the country. The integration of the National ID (NID) system guarantees that only the qualified or genuine voters can participate in the election process eliminating the risk of imposters or multiple votes. In order to keep voters' identities private while logging in, Ring signatures mix the NID data in a way that guarantees legitimacy without revealing who they are. The system is built with accessibility in mind, offering a responsive web application, ensuring it is accessible across a wide range of devices including desktops, tablets, and smartphones. Election authorities and administrators benefit from built-in auditing features such as zk-SNARK verification and Merkle Tree-based hashing facilitating real-time monitoring without accessing confidential voter data. Therefore, this framework guarantees smooth and secure voting processes across a wide range of user demographics and operational conditions.

iii. Economic Feasibility

From an economic standpoint, the DeVote system offers a cost-effective alternative to both traditional voting methods and blockchain-based platforms that rely on third-party networks. Building the custom blockchain helped the project prevent typical fees for networks like Ethereum (e.g. gas fees), making it more practical for big elections. Even though initial investment for the system can be significant at first, the system lowers operational costs in the long run by eliminating expenditures on ballots, facilities, staff and logistical work. Additionally, decentralized architecture reduces the need for continuous manual oversight and centralized maintenance. Thus, this project is highly economic because it decreases the need for middlemen and offers governments in low-income countries a budget-friendly alternative for holding secure and verified elections digitally.

iv. Schedule (Gantt chart showing the project timeline)

Project Schedule for DeVote (May-August 2025)

Task	May			June				July				August			
	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4
Requirement Identification															
Feasibility Study															
System Design															
Prototype Development															
System Integration															
Testing & Debugging															
Final Review & Documentation															

Figure ii: Project Timeline for Devote (May-August, 2025)

c. High-Level System Design

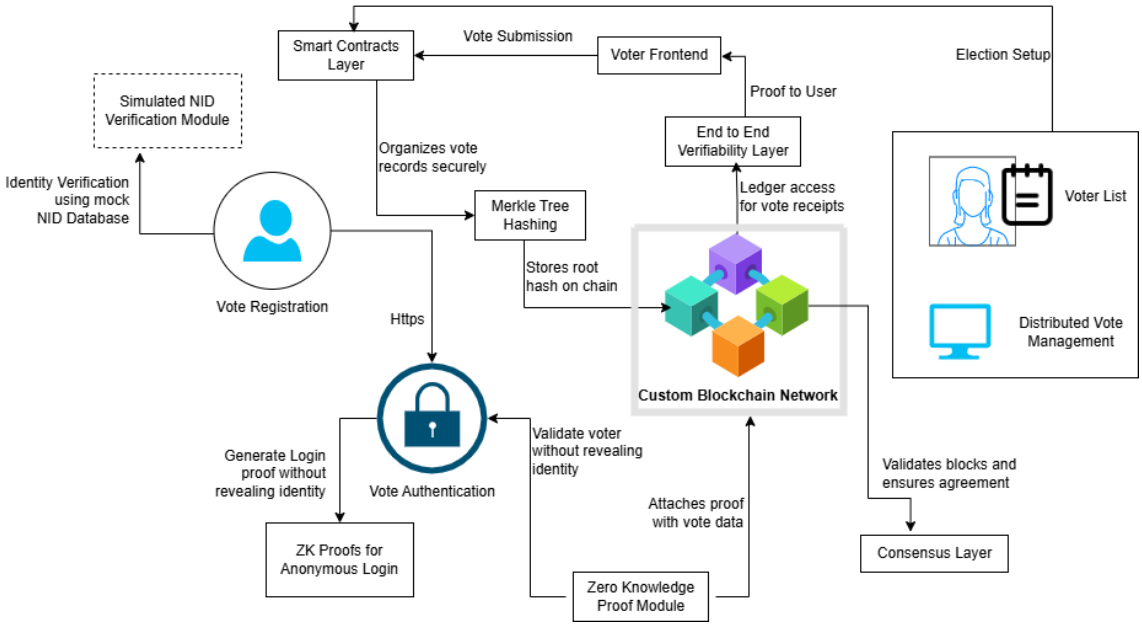

Figure iii: Architecture of the DeVote system

i. Methodology of Proposed System

To build the DeVote system, we've decided to follow the Agile Incremental Model. This approach lets us break down the whole system into smaller parts and develop them step by step. After completing each part, we test it, improve it, and then move on to the next. This way, we're not overwhelmed by the complexity of the whole project at once, and we can keep improving as we go.

One of the biggest reasons we chose Agile is because of its flexibility. Since we're working on a new concept like blockchain-based voting, things can change as we learn more or get feedback. Agile makes it easier for us to adapt to those changes without needing to start everything over. For example, if we realize midway that a certain part of the user interface can be made more secure or user-friendly, we can just adjust that part in the next cycle.

Our plan is to build the system gradually — starting with the basics like user registration and login, and then moving on to more complex features like secure voting, blockchain integration, and real-time result display. After each feature is developed, we'll test it thoroughly to make sure it works well and meets our goals.

Throughout the process, we'll also hold regular team meetings to track our progress, discuss any challenges, and make decisions together. This ensures that everyone stays on the same page and that the system we're building reflects what users actually need. Therefore, the Agile method helps us stay organized, flexible, and user-focused.

Working Mechanism of the Proposed System

1. User Registration and Authentication: At first, voters register on the system by providing the credentials from the NID Card. To ensure legitimacy, the system verifies user data against a preloaded or linked database. Once verified, users can log in securely using their credentials.

2. Election Setup by Admin: The admin sets up the election details such as election title, date and time range, list of candidates, and voting rules. This part is only accessible to authorized personnel with admin rights.

3. Blockchain-Based Voting Process: Once the voting starts, verified voters can log in and cast their vote. Each vote is immediately encrypted and added to a blockchain network. Blockchain ensures that once a vote is recorded, it cannot be altered, deleted, or tampered with, preserving both security and transparency.

4. Vote Confirmation & Anonymity: After casting the vote, the user receives a confirmation. While votes are stored securely on the blockchain, the system ensures voter anonymity meaning no one can trace back which user voted for which candidate. This maintains privacy and fairness.

5. Result Calculation and Display: Once the voting time ends, the system automatically tallies the votes stored on the blockchain. Since the data is immutable and transparent, the results are trustworthy. The final outcome is displayed in a user-friendly format for both admin and general viewers.

6. Audit and Verification: For highly secure or public elections, an audit trail can be generated using Merkle Trees. This allows validators to verify the integrity of the data without exposing sensitive information. It adds another layer of transparency and builds trust in the system.

To conclude, DeVote follows a straightforward yet secure process: voters register, vote securely through blockchain, and view accurate, tamper-proof results with the help of an easy-to-use interface.

ii. **Flowchart of the Proposed System**

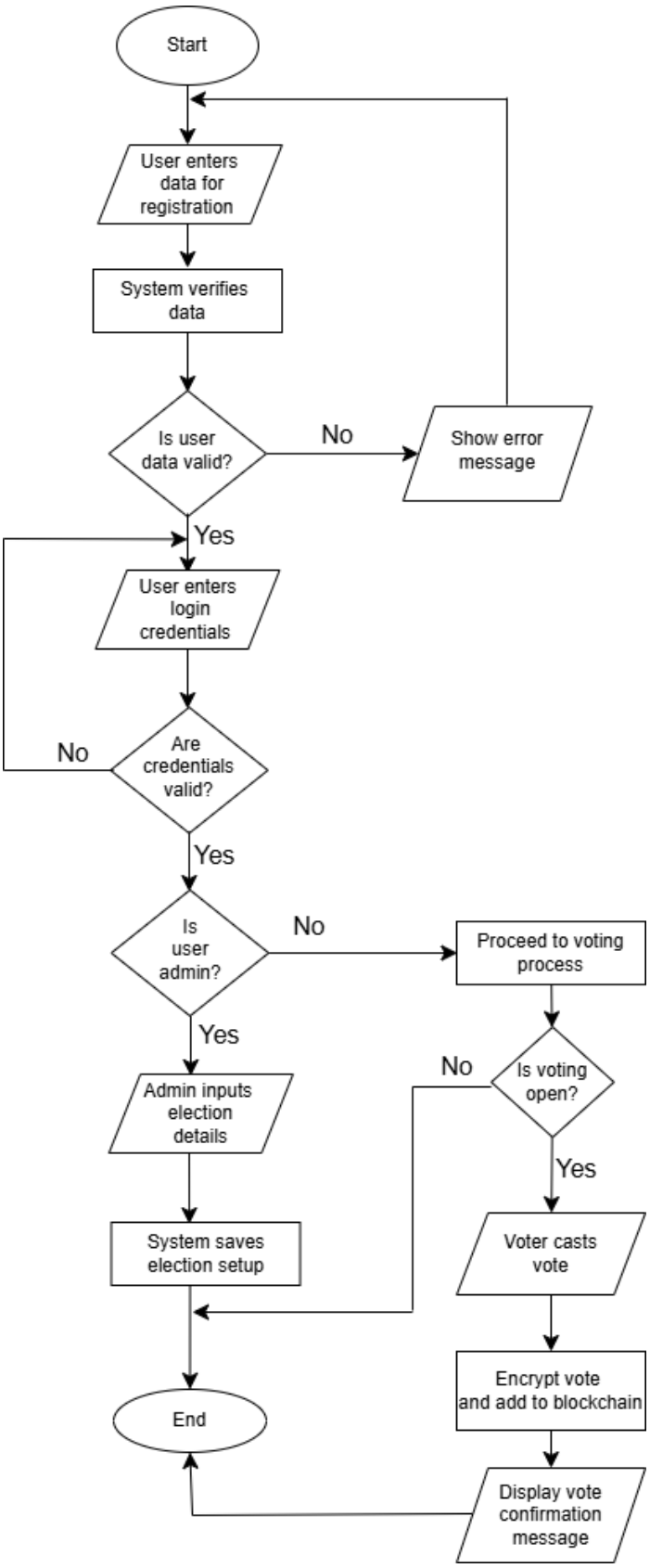

Figure iv: Illustration of working mechanism of DeVote in a flowchart

iii. Description of algorithm

SHA-256

SHA 256 is one of the main components of the blockchain technology which is used to create a unique output (relatively known as digest, or message digest) for an input having l bits, where $0 \leq l \leq 2^{64}$. Cryptographic Hash functions have some important properties, which includes

1. Pre-image resistant (One-way function): Hash functions are one-way functions, which means calculating the input based on the output is computationally infeasible. For instance, given a digest, find x such that hash(x) = digest is not computationally possible.
2. Second pre-image resistant: It means it is not possible to find a different input for a specific output. Given x, finding y such that hash(x)=hash(y) is not computationally feasible.
3. Collision Resistant: It means we cannot find two different inputs that have same output digest. Finding an x and y such that hash(x) = hash(y) is not feasible.

For this purpose, we will be using SHA (Secure Hash Algorithm) 256 algorithm that has all the above properties. SHA 256 takes an input of any size and converts it into a fixed output of 32 bytes (256 bits). SHA 256 will be used to create unique identifiers for each block in the blockchain [4].

SHA-256 algorithm uses six logical functions which are,

- $\mathrm{Ch}(\mathrm{x},\mathrm{y},\mathrm{z}) = (\mathrm{x} \wedge \mathrm{y}) \oplus (\neg \mathrm{x} \wedge \mathrm{z})$
- $\mathrm{Maj}(\mathrm{x},\mathrm{y},\mathrm{z}) = (\mathrm{x} \wedge \mathrm{y}) \oplus (\mathrm{x} \wedge \mathrm{z}) \oplus (\mathrm{y} \wedge \mathrm{z})$
- $\sum_0(x) = ROTR^2(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x)$
- $\sum_1(x) = ROTR^6(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x)$
- $\sigma_0(x) = ROTR^7(x) \oplus ROTR^{18}(x) \oplus SHR^3(x)$
- $\sigma_1(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x)$

ALGORITHM:

1. Preprocessing: It consists of 3 steps, which are padding the message, parsing the message into blocks, and setting the initial hash value.
    a. Padding the message: Add a single 1 bit followed by 0 bits to make the length of the message congruent to 448 mod 512. Finally, append the original message length l as a 64-bit binary number. Hence, the padded message becomes a multiple of 512 bits.

    $$\text{Padded Length} \equiv 448 \pmod{512}$$

    b. Parsing the message: The padded message is split into 512-bit blocks.

    $$\text{Padded message} = 512\text{-bit} * \text{N blocks}$$

    c. Setting the Initial Hash Value: The initial hash value of the SHA-256 consists of following eight 32-bit words.

$$H_0^{(1)} = 6a09e667$$
$$H_1^{(1)} = bb67ae85$$
$$H_2^{(1)} = 3c6ef372$$
$$H_3^{(1)} = a54ff53a$$
$$H_4^{(1)} = 510e527f$$
$$H_5^{(1)} = 9b05688c$$
$$H_6^{(1)} = 1f83d9ab$$
$$H_7^{(1)} = 5be0cd19$$

2. Hash Computation: It uses the six previously defined logical functions. Each message block, $M^{(1)}$, $M^{(2)}$, ……, $M^{(n)}$ processed through the following steps.
    a. Prepare message Schedule: Create 64 words W_0 *to* W_{063}, each consisting of 32 bits from each 512-bit block.
        - For t=0 to 15:

        $$W_t = From\ the\ current\ message\ block$$

        - For t=16 to 63:

        $$\sigma_0(x) = ROTR^7(x) \oplus ROTR^{18}(x) \oplus SHR^3(x)$$
        $$\sigma_1(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x)$$
        $$W_t = \sigma_1(W_{t-2)} + W_{t-7} + \sigma_0(W_{t-15)} + W_{t-16}$$

b. Compression Function

- Initialize the eight working variables with the $(i-1)^{st}$ hash value.

$$a = H_0^{(i-1)}$$
$$b = H_1^{(i-1)}$$
$$c = H_2^{(i-1)}$$
$$d = H_3^{(i-1)}$$
$$e = H_4^{(i-1)}$$
$$f = H_5^{(i-1)}$$
$$g = H_6^{(i-1)}$$
$$h = H_7^{(i-1)}$$

- For t=0 to 63:

$$T_1 = h + \sum_1(e) + Ch(e,f,g) + K_t + W_t$$

$$T_2 = \sum_0(a) + Maj(a,b,c)$$

h=g

g=f

e=d+T_1

d=c

c=b

b=a

a=$T_1 + T_2$

c. Compute the i[th] intermediate hash value H[(i)]:

$$H_0^{(i)} = a + H_0^{(i-1)}$$
$$H_1^{(i)} = b + H_1^{(i-1)}$$
$$H_2^{(i)} = c + H_2^{(i-1)}$$
$$H_3^{(i)} = d + H_3^{(i-1)}$$
$$H_4^{(i)} = e + H_4^{(i-1)}$$
$$H_5^{(i)} = f + H_5^{(i-1)}$$
$$H_6^{(i)} = g + H_6^{(i-1)}$$
$$H_7^{(i)} = h + H_7^{(i-1)}$$

After repeating steps one to four a total of N times, the resulting 256-bit message digest of the message, M, is

$$H_0^{(N)} \parallel H_1^{(N)} \parallel H_2^{(N)} \parallel H_3^{(N)} \parallel H_4^{(N)} \parallel H_5^{(N)} \parallel H_6^{(N)} \parallel H_7^{(N)}$$

RSA Algorithm

RSA (Rivest-Shamir-Adleman) is a widely used public-key cryptosystem that enables secure data transmission through asymmetric encryption. It relies on a pair of keys: a public key (used for encryption) and a private key (used for decryption). RSA ensures confidentiality, authenticity, and integrity in digital communications.

Key Generation:

1. Select two large prime integers p, q such that p $\neq q$.
2. Calculate the modulus

   n = p*q.
3. Calculate Euler's totient function

   $$\phi(n) = (p-1)(q-1).$$
4. Select integer e such that $\phi(n)$ and e are relatively prime,

   $$\text{Gcd}(\phi(n), e) = 1\ ,\ 1 < \text{e} < \phi(n).$$
5. Calculate d $\quad d \equiv e^{-1}(mod\ \phi(n))$
6. Public Key PU = {e, n}
7. Private Key PR = {d, n}

Encryption:

1. Plaintext: M<n, where M is message plaintext is ASCII.
2. Ciphertext: C = $M^e\ mod\ n$.

Decryption:

1. Ciphertext: C
2. Plaintext: M =$C^d mod\ n$.

5. EXPECTED OUTCOME

The project will deliver a secure blockchain-based voting system where votes are immutably recorded using blockchain technology, ensuring tamper-proof integrity. Advanced cryptography (zk-SNARKs and Ring Signatures) will guarantee voter anonymity while enabling verifiable participation. Through Merkle trees, voters can confirm their vote was counted without compromising secrecy. The system will feature a user-friendly interface for universal accessibility and integrate NID-based authentication with zero-knowledge proofs to validate eligibility privately. PBFT consensus will ensure efficient, decentralized validation, making the process resistant to fraud. By combining these features, the solution will provide a transparent, auditable, and accessible voting platform that enhances trust, security, and participation compared to traditional methods.

6. REFERENCES

[1] D. Yaga, P. Mell, N. Roby, K. Scarfone, "Blockchain Technology Overview," *National Institute of Standards and Technology,* vol. 1, 10 2018.

[2] "Follow My Vote, “Online Blockchain Voting,”".

[3] G. Wood, "Ethereum: A Secure Decentralised Generalised Transaction Ledger," 2014.

[4] Willie E. May, "Secure Hash Standard (SHS)," *FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION,* 2015.

[5] D. Mahto and D. Kumar, "RSA and ECC: A Comparative Analysis," *International Journal of Applied Engineering Research,* vol. 12, pp. 9053-9061, 2017.

[6] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.

[7] B. Adida, "Helios: Web-based Open-Audit Voting," 2008.

[8] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized Anonymous Payments from Bitcoin," 2014.