

**THE TECHNICAL UNIVERSITY OF KENYA  
FACULTY OF APPLIED SCIENCES AND TECHNOLOGY**

**SCHOOL OF COMPUTING AND INFORMATION TECHNOLOGY**

**COMMUNICATION AND COMPUTER NETWORKS**

STUDENT NAME: **SAMUEL MBUGUA KARANJA**

ADM NO: **SCNI /OO445/2021**

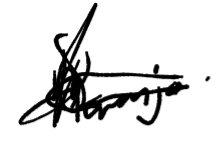
**TITLE:**

**A SECURE AND TRANSPARENT DECENTRALIZED WEB VOTING SYSTEM FOR ELECTIONS IN KENYAN ORGANIZATIONS**

PROJECT A SUBMITTED TO THE SCHOOL OF COMPUTING AND INFORMATION TECHNOLOGY IN PARTIAL FULFILLMENT FOR THE BACHELOR OF COMMUNICATION AND COMPUTER NETWORKS OF THE TECHNICAL UNIVERSITY OF KENYA

# **DECLARATION**

I, SAMUEL MBUGUA KARANJA, admission number SCNI/00445/2021, a final year student at TECHNICAL UNIVERSITY OF KENYA hereby certifies that the work which is being presented in the project is an authentic record of my own work and has been developed as part of my academic requirement. I can confirm that this project work is fully my idea and hasn’t been submitted elsewhere for academical or other purposes.

Samuel Mbugua Karanja   
SCNI/00445/2021  


**Supervisor**

**Supervisor 1  
Name:** Luke Okelo **Signature:   
  
Supervisor 2  
Name:   
  
Signature:**

# **DEDICATION**

**I** dedicate this project to my family, friends, mentors and lecturers who have been constantly supporting and encouraging me throughout my work. Lastly, I thank my God for giving me the strength, wisdom, the knowledge and grace. Without Him this would have been impossible.

# **ACKNOWLEDGEMENT**

First and foremost, I express my deepest gratitude to sir God for him giving me full strength, wisdom and perseverance to push my project from one point to another.

I would also like to thank my supervisors: Mr Luke Okello and Madam Dorcas Awino Ebbie for their invaluable guidance, timely feedback and continuous support throughout my project. Your mentorship has been instrumental in coming to understand blockchain technology and its application.

I am also grateful to my lecturers and academic advisors all around me for sharing their knowledge and also encouraging me to develop skills necessary for my project.

To my family and friends, you aren’t forgotten as you fully supported and motivated me during this challenge. Your belief in me was a great inspiration.

Lastly, I do appreciate everyone who contributed to this project directly or indirectly you were a joy to have around.

# **LIST OF ABBREVIATIONS**

# **LIST OF FIGURES**

# **LIST OT TABLES**

# **ABSTRACT**

Elections in Kenyan organizations such as NGOs, SACCO, private and public institutions often face challenges while performing their voters right in their organizations e.g. voter fraud, lack of transparency, human errors and delayed results which compromises their democratic integrity and trust. This project proposes a secure, transparent decentralized web voting system by use of blockchain technology to address these issues.

The system is aiming to ensure tamper-proof vote recording, real-time tallying, and user-friendly access to voters under a small-scale organizational election. It uses blockchain technology to securely record votes, ensuring immutability and transparency. The proposed system will introduce features for user registration, authentication and real-time vote tallying with levels for access i.e. normal users(voters) and administrators. The system will implement off-chain operations for non-critical actions while utilizing layer 2 scaling solutions. This will aid in reducing the congestion and transaction costs. The user-friendly interface will ensure that the persons with minimal technical knowledge can still participate effectively for it will be simple and easy to understand.

The project will follow Agile Evolutionary methodology that will ensure that by the end of the project I will have developed a well functional prototype that will incorporate voter authentication, smart contracts and an off-chain database. Data collected from willing participants will be in form of interviews (taking detailed points from the interviewees), questionnaires(online) and a simulated election (for testing the system). The expected output is a fully functional decentralized voting system that will guarantee one person one vote while providing end to end verifiability which minimizes electoral fraud. The system will also support a transparent vote auditing and secure voter identity management. Ultimately, the project offers an adaptable model that can be utilised in organizational settings and other institutions to improve democratic processes through blockchain technology.

**Table of Contents**

[**DECLARATION** ii](#_Toc200612345)

[**DEDICATION** iii](#_Toc200612346)

[**ACKNOWLEDGEMENT** iv](#_Toc200612347)

[**LIST OF ABBREVIATIONS** v](#_Toc200612348)

[**LIST OF FIGURES** vi](#_Toc200612349)

[**LIST OT TABLES** vii](#_Toc200612350)

[**ABSTRACT** viii](#_Toc200612351)

[**CHAPTER 1: INTRODUCTION** 1](#_Toc200612352)

[o **Introduction** 1](#_Toc200612353)

[**1.2 Background of the study** 2](#_Toc200612354)

[1.2.1 Background 2](#_Toc200612355)

[1.2.2 Overview of existing system 3](#_Toc200612356)

[1.2.3 Overview of the blockchain-based system 4](#_Toc200612357)

[**1.3 Problem statement** 6](#_Toc200612358)

[**1.4 Main Objectives** 7](#_Toc200612359)

[1.4.1 project goal 7](#_Toc200612360)

[**1.5 Justification** 8](#_Toc200612361)

[**1.6 scope of the study** 9](#_Toc200612362)

[**1.7 limitation of decentralized voting systems** 10](#_Toc200612363)

[**1.8 project risks and mitigation** 11](#_Toc200612364)

[**1.9 Project schedule** 12](#_Toc200612365)

[**Time schedule** 12](#_Toc200612366)

[**1.10 budget and resources** 14](#_Toc200612367)

[**CHAPTER 2: LITERATURE REVIEW** 15](#_Toc200612368)

[**2.1 Reviewed similar systems** 15](#_Toc200612369)

[**2.2 Tools and methodologies** 19](#_Toc200612370)

[**2.3 Gaps in the existing systems and the proposed solution** 19](#_Toc200612371)

[**2.4 The proposed solution** 20](#_Toc200612372)

[**CHAPTER 3: METHODOLOGY** 21](#_Toc200612373)

[**3.1 Methodology and Tools** 21](#_Toc200612374)

[**3.2 Source of Data** 22](#_Toc200612375)

[**3.3** **Data Collection Methods** 22](#_Toc200612376)

[**3.4 resources and materials required** 25](#_Toc200612377)

[**CHAPTER 4: SYSTEM ANALYSIS AND REQUIREMENT MODELING** 26](#_Toc200612378)

[**4.1** **Introduction to System Analysis** 26](#_Toc200612379)

[**4.2** **Objectives of System Analysis** 26](#_Toc200612380)

[**4.3** **Problem Definition** 26](#_Toc200612381)

[**4.4** **Feasibility Study** 27](#_Toc200612382)

[**4.5** **System Analysis Tools** 27](#_Toc200612383)

[**4.6** **System Investigation** 28](#_Toc200612384)

[4.6.1 Introduction 28](#_Toc200612385)

[4.6.2 Data Collection Methods 28](#_Toc200612386)

[4.6.3 Fact Recording 28](#_Toc200612387)

[**4.7** **System Analysis** 29](#_Toc200612388)

[4.7.1 Current System Analysis 29](#_Toc200612389)

[4.7.2 Proposed System Analysis 32](#_Toc200612390)

[4.7.3 Requirement Definitions and Specifications 38](#_Toc200612391)

[4.7.4 Comparison 38](#_Toc200612392)

[**CHAPTER 5: SYSTEM DESIGN** 40](#_Toc200612393)

[**5.1 Introduction to System Design and Nature of the System** 40](#_Toc200612394)

[**5.2** **Objectives of System Design** 40](#_Toc200612395)

[**5.3** **Program Design Tools** 41](#_Toc200612396)

[**5.4** **Logical Design** 41](#_Toc200612397)

[5.4.1 Logical Data Design: Relational Data Analysis / Normalization 41](#_Toc200612398)

[5.4.2 Entity Attributes Relationships 42](#_Toc200612399)

[**References** 43](#_Toc200612400)

[**Appendix** 43](#_Toc200612401)

# **CHAPTER 1: INTRODUCTION**

## **Introduction**

In recent years, often after elections are conducted and tallied, what follows after is always the losing party reporting electoral fraud or bring something up which leads to the system used during that election being the ‘bad guy’. This leads to violence in various occasions, leaders getting seats which were meant for in various organizations and many more inconveniences which leads to demand of an efficient, secure and transparent voting system. We have seen cases like these on tv reports, national newspapers, social media and history books. This is due to the traditional voting systems lacking transparency, having security vulnerabilities and sometimes logical errors. To address these challenges, blockchain technology provides a decentralized and immutable way to conduct elections efficiently. This project aims to develop a blockchain-based voting system that ensures and guarantees security, transparency and accessibility.

Why do we vote? People vote because they want to exercise their right to influence decisions that affect them at their day-to-day life at their respective organizations, institution, nations, regions and also globally at a full. They want to vote because they feel they need to:

* **Be included**- the people get to vote a leader who will represent their say when it comes to decisions affecting them and their surroundings
* **Be accountable-** holds decision -makers accountable, allowing people to reward them based on performance. Either upgrade or degrade them according to their performance status.
* **Resolve the conflict among themselves**- as they will always be an opposing side, voting systematically allows people to peacefully resolves disputes accordingly, the majority vote wins automatically.

What if we don’t vote? When people don’t vote this may cause tension as they feel they aren’t involved in the general matters. This will lead to:

* **Unrepresentative outcome-** the people may get decisions that may go against them from the ruling class.
* **Disengagement**- there might be disengagement between the leaders and the people.

What do we get from voting?

* **Inclusivity-** the people feel represented by their respective leaders when it comes to making decision regarding them
* **Transparency and accountability**- voting holds decision makers answerable to the people reducing mismanagement.
* **Improved decision quality-** diverse participation brings to varied perspectives leading to better informed outcomes for the people.
* **Social unity**- voting unites people around shared goals strengthening community bonds.

## **1.2 Background of the study**

### 1.2.1 Background

***what is voting****?* It is the will to express or signify a person’s choice in a matter by any means available to them.  
Types of voting in Kenyan organizations include:

1. **Paper ballot voting**- traditional voting where individuals cast votes on paper and drop them in a ballot box.
2. **Electronic voting(e-voting)-** use of electronic machines to cast and count votes.
3. **Online voting-** digital voting systems which are accessed via the internet, often used on public organizations.

**ICT Adoption in Voting**

The use of ICT in voting in Kenya began in the early 2000s, with major organizations adopting electronic systems to improve efficiency. Institutions such as independent electoral and boundaries commission (IEBC) have explored in digital solutions for more efficiency. It was until 2010s when web-based systems were introduced to facilitate transparency and efficiency. (Micheni, 2018), The role of ICT in electoral processes)

Why conduct elections? Elections play a crucial part when it comes to leadership, especially in democratic governance to ensure that the people have a voice in selecting their preferred leader. However, the traditional voting system often faces challenges before, during or after elections like voters’ fraud, lack of transparency and human errors which leads to having wrong leaders getting undeserved merits. With technology on the rise, blockchain-based voting systems have emerged as alternate solution to the traditional system to address these challenges. By leveraging blockchains immutability and cryptographic security, this system is aiming to eliminate electoral fraud on large scale.

The voting historically in our county Kenya as a multiparty system was introduced in 1991 then conducted on 1992. (Bratton, 2008), voting in Kenya: putting ethnicity in perspective). The Kenyan republic started enjoying to have the rights to participate in democratic processes while interest in governance increased. Nonetheless one thing remained a constant during the election process- (Mebane, 2011)voter fraud, as Allegations of electoral malpractices and unfairness emerge after every general election. The organizational voting system was introduced later to promote democratic rights within institutions in the late 1990s.

blockchain voting in KenyaCurrently, no major organizations in Kenya have fully adopted blockchain-based voting systems, however there are some research and pilot projects underway to explore its potential in enhancing electoral integrity. Example of some major blockchain voting systems events in Kenya include:

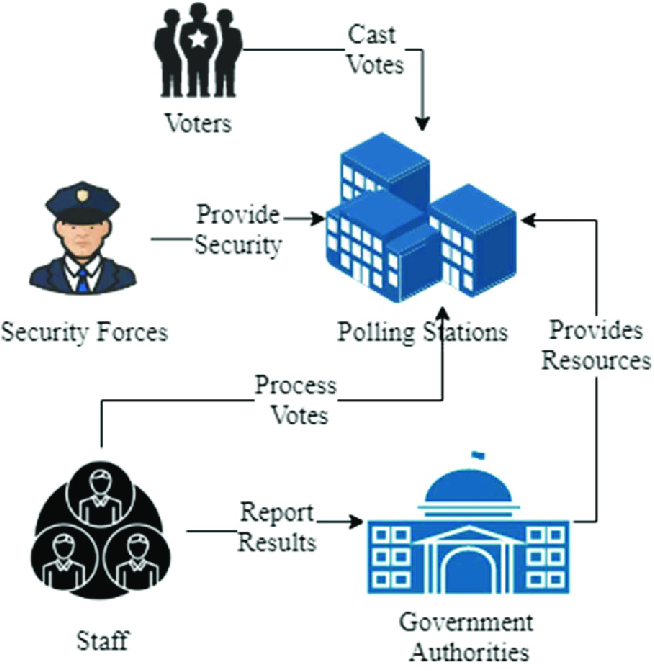
* IEBC proposed blockchain in 2018 for transparent result transmission but was not enacted. (Wafula, 2018) Crypto in Africa: Kenya’s electoral body seeks to adopt blockchain)

### 1.2.2 Overview of existing system

The current systems mostly are electronic, paper based (use of ballot paper) in some occasions or both applied, but are centralised bringing up various operational challenges. The people visit the polling station with their voting cards which is needed and identity card. Their biometrics are taken for more verification. The traditional system involves some few procedures:

1. **Voter registration**-people register at the polling station and must be present physically or use online systems which require manual verification.
2. **Ballot casting**- the voters visit the polling station to cast their votes or use centralised electronic voting systems which are vulnerable to manipulation and hacking.
3. **Vote counting**- votes are counted manually which may lead to human error and/or potential fraud.
4. **Result announcement**-elections are announced after some few hours or days which might face disputes over credibility.

***Fig 1.1: electronic voting process***



**Problems faced by the existing system**

* **Voter fraud-** the system is susceptible to identity fraud, vote duplication and vote manipulation. (vorobyev, 2013) Essays on electoral fraud explains fraud as it happens during election.
* **Slow and costly-** manual vote counting and verification may delay results and increase the cost. (Harris, 2021) election administration, Resource Allocation and Turn out
* **Security-** the votes are vulnerable to cyber attack in the centralised electronic system which also the manually counted votes are filled in the systems which might be hacked.
* **Transparency-** the centralised and manual nature of the systems limits public **verification of results.**

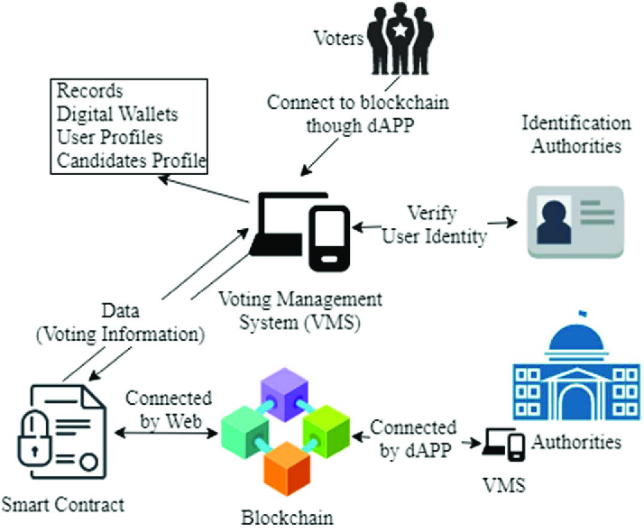
### 1.2.3 Overview of the blockchain-based system

The idea of decentralised voting system will leverage blockchain technology to enhance security, transparency and efficiency.  
The key steps to be followed include:

1. **Voter registration-**voters will register on a blockchain network, this is to ensure immutability and verification of voters without risk of tampering.
2. identity verification- registered voters have to be verified through national id, biometrics or any government issued documents.
3. eligibility- confirmed through zero-knowledge proof to maintain privacy.
4. **Smart contract ballot casting**-the voters will cast their votes digitally on an encrypted blockchain based platform (still figuring out the best platform)
5. Voters to receive a unique key
6. The platform will ensure anonymity and encryption
7. Smart contracts validate and record each vote on the blockchain in real-time
8. **Automated vote counting**
9. Votes are counted as they cast.
10. Decentralized ledger helps in ensuring no vote duplication or alteration
11. **Instant Result announcement**- Since the system operates on real-time, vote tallying and results will be publicly verifiable without any party or intermediary manipulating the data by the end of counting.

Multi-party verification will ensure trust in the outcome.

*fig 1.2: steps in blockchain voting process*



**Benefits of blockchain-based system**

* **Enhanced security-** blockchain technology ensures votes are immutable and tamper-proof.
* **Increased transparency-** every vote is recorded on a public ledger for easier verification by parties involved.
* **Cost efficient-** e.g. manual vote casting which requires human labour on the tradition systems.
* **Faster results-** since vote counting is automated, this eliminates delays for results announcement.

## **1.3 Problem statement**

The current electoral system faces various challenges that affect the integrity and efficiency of elections. The challenges arise due to how some group of people want a specific person to lead them because they might get some favours under his reign thus leading to election inefficiencies, security challenges and lack of transparency during the election process. All of this raise’s concerns to the majority people.

The key problems include:

* Voter fraud- the existing system relies more on manual work e.g. manual verification which is prone to duplicate registrations, identity theft and unauthorized voting which leads to fraudulent outcomes.
* Lack of transparency- it may occur during vote counting and result announcement. The traditional system votes tallying faces lack of transparency creating opportunities for vote manipulation and errors. The lack of transparent publicly verifiable processes, reduces voters’ confidence in the outcome of the elections.
* Security threats and cyber vulnerabilities- the centralised databases systems are prone to cyber-attacks, unauthorised logins and data breaches. The lack of decentralised system leaves the data(votes) to manipulation.
* Delays and high cost in election management- manual processes in voter registration, ballot casting and results computation leads to a prolonged election period hence increased cost. The inefficiencies in the entire process make it costly and time consuming.

To address these issues, this project proposes the development of a blockchain-based decentralised system. By using blockchain technology, this system aims to provide a transparent, efficient and tamper-proof alternative to the traditional voting methods. It will enhance security through cryptographic verification, ensure fast results due to the real-time results tallying and public auditability thereby gaining public trust.

## **1.4 Main Objectives**

### 1.4.1 project goal

To develop a secure, transparent and decentralised blockchain-based voting system that ensures the highest level of election integrity, prevents electoral fraud and enhances voters’ confidence.

1.4.2 specific objectives

1. To develop a web-based blockchain voting system that ensures secure, transparent, and tamper-proof elections.
2. To implement a secure voter authentication module that prevents double voting and ensures only eligible voters participate.
3. To design and integrate a smart contract that securely records, stores, and verifies votes on the blockchain.
4. To build a decentralized vote tallying system that ensures accurate and real-time election results.
5. To create a user-friendly voting interface that simplifies vote casting and result verification.

## **1.5 Justification**

The integrity of the electoral process is fundamental in any democracy or organisation, yet elections continue to face challenges. The proposed blockchain-based decentralised voting system presents an innovative solution that enhances security, transparency and efficiency to the electoral process.   
This project aims to create a fully decentralized blockchain system that will address these challenges in various ways.  
The system will have an automated vote tallying to ensure instant results doing away with the old way of manual counting. The total cost of the operation will be reduced by removing several expenses e.g. ballot box, administration costs and manual verification.  
The system will introduce some tight security measures e.g. cryptographic encryption and blockchain immutability to prevent vote tampering and any fraudulent activity.  
It will also allow remote voting in order to increase accessibility and voter participation. This will lead to great number turn out.   
Through real-time blockchain access, the system aims to save time for election observers and organizers at the same time making the elections easier to monitor.

## **1.6 scope of the study**

This project focuses on designing a decentralised voting system using blockchain technology to enable transparent and secure elections within a small organization. The study will cover the development and implementation of the system to demonstrate key functionalities.

In-scope

1. The blockchain system will allow both administrators and voters to log in securely.
2. The system will enable administrators to register voters and create new voting accounts.
3. The system will provide a smart contract mechanism to securely store and verify votes on the blockchain.
4. The system will facilitate real-time vote tallying using decentralized ledger technology.
5. The system will implement an authentication module to prevent double voting and unauthorized access.
6. The system will provide an intuitive user interface for casting votes and verifying election results.
7. The system will include an off-chain database for storing non-critical data such as voter registration details.

Out-scope

1. The system will not integrate with physical voting hardware. This is due to the focus on a software-based solution.
2. The system will not send voter update messages or notifications.

## **1.7 limitation of decentralized voting systems**

While the decentralised voting systems offers a lot of advantages, it also faces some several limitations. They include:

1. The system will not allow voters to change their votes once submitted
2. The system will not provide multi-device synchronization for voting sessions.
3. The system will not include real-time election monitoring dashboards for external observers.
4. The system will not support voting through SMS or email.

## **1.8 project risks and mitigation**

The development and implementation of the blockchain-based decentralised voting system faces some several risks.

Project risk: Security breaches-there may be potential smart contract vulnerabilities.  
project mitigation: conduct thorough security audits and testing.

Project risk: system vulnerability to hacks -online voting may be vulnerable to attacks from hackers as they say every system is hackable.

Project Mitigation: ensuring the system is security fit and explore offline voting alternatives for future versions.

Project risk: User adoption-final users may struggle to adapt with blockchain technology  
Project mitigation: provide user friendly UI and guides.

Project risk: Data breaches-there maybe unauthorized access to voting records.  
Project mitigation: implement a robust encryption and access control.

## **1.9 Project schedule**

### **Time schedule**

Project timeline  
The project has a month’s timeline from project proposal phase to full completion. In order to achieve that, the project was divided into various phases in order to finish each phase at a particular point.

* Phase 1 (2 weeks)  
  This involves project planning and research. It took exactly two weeks to gather the information about existing systems.
* Phase 2 (4 weeks)  
  This phase involves system designing. Due too much filling and thinking this phase consumed much time. Some work done include:  
  1. Creating data flow diagrams for showing information flow in the system.
  2. Designing User Interface
  3. developing Entity Relationship diagrams for showing the relationship.
* Phase 3 (7 weeks)  
  System development. It involves building the system from the foundation to connecting different activities within the system. Activities include:
  1. Backend development
  2. frontend development
  3. coding
  4. smart contracts application etc
* Phase 4 (3 weeks)

It involves system testing and optimization.  
In this phase some several system test runs will be done to make sure the system is well responsive and errors will be fixed accordingly. Some activities will include:

* 1. system integration.
  2. perform some run test on the system to observe any errors.
  3. error and bugs fixing
* Phase 5 (3 weeks)

Introduce the system to some users for them to engage with it and receive some feedback from them.  
will address issues or concerns raised also

* Phase 6 (2 weeks)  
  System deployment and documentation.

This will involve system deployment for organizational use while giving out the full documentation for usage and maintenance.  
Do some final performance test to check if the system is ready for use.

## **1.10 budget and resources**

The development of the decentralised voting system requires several resources (hardware, software and human resources) to achieve the project objectives. My budget and resource plan prioritizes readily available, low cost or free tools to ensure full completion.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Resources | description | specifications | Price (KES) | units |
| Hardware resources | Laptop, computer | 8GB RAM 256GB ROM modern processor | Approx. 25,000 | 1 |
| Software resources | Dev tools-vs code blockchain platforms  documentation tools-Ms word | Latest versions. | free | n/a |
| internet | Reliable internet connection | Fast network | 900/- month | n/a |
| Human resource | Support and guidance | Peers and supervisor guidance | n/a | n/a |
| others | Backup and storage | USB drive | 500 | 1 |

# **CHAPTER 2: LITERATURE REVIEW**

This chapter presents an overview of related works, in this case field related work in electronic and blockchain-based voting systems. It critically examines and analyses similar systems, their tools and methodologies, and their limitations. I have done some research on the same with the help of google scholar, web search and generative AIs.

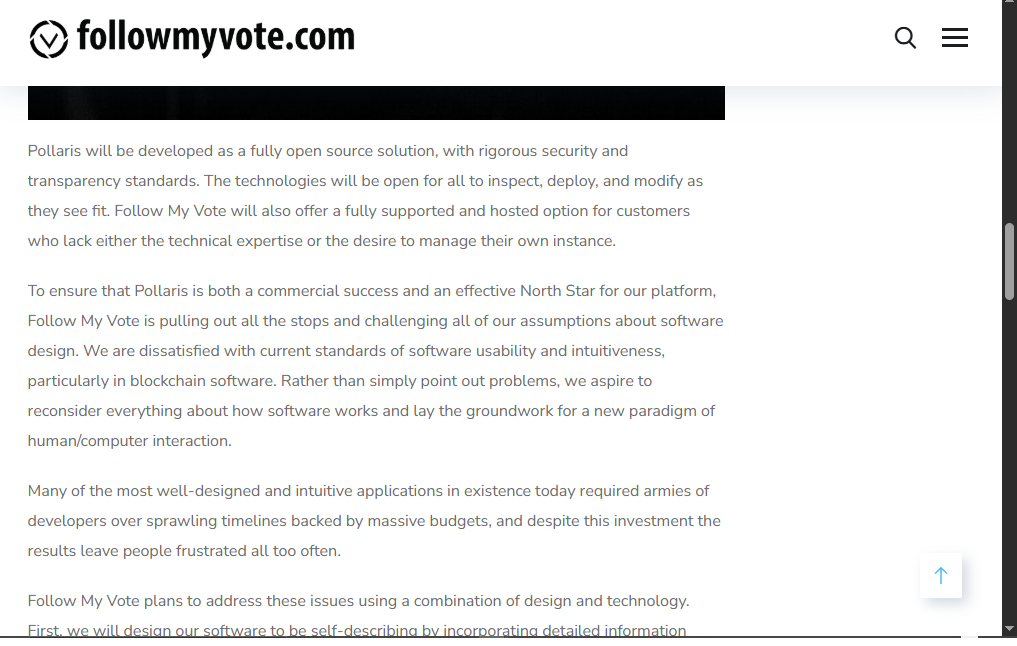
## **2.1 Reviewed similar systems**

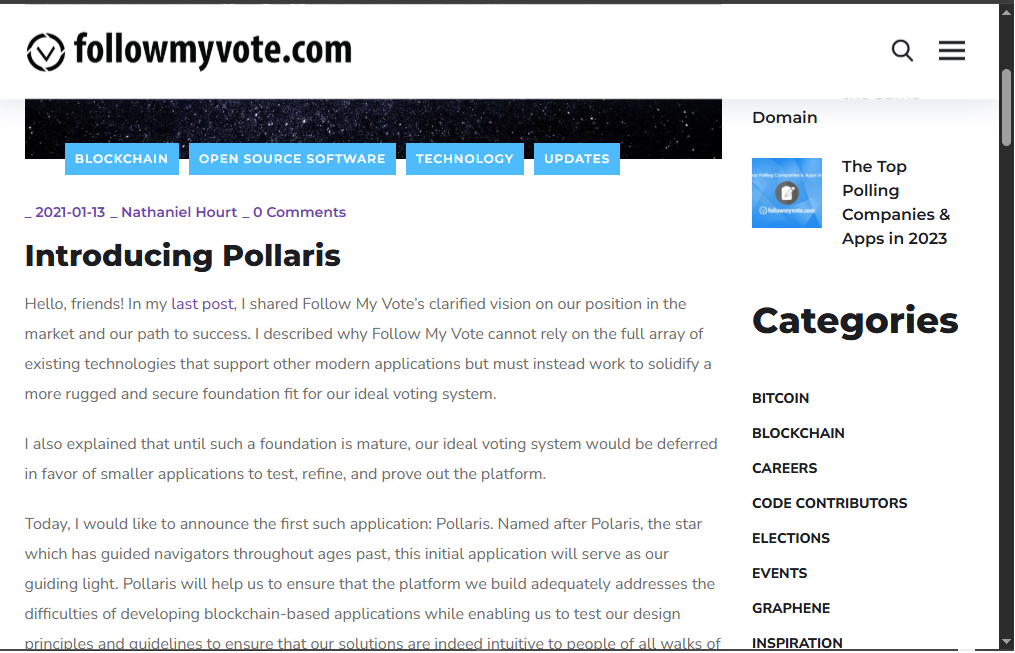
Several similar systems have been developed to address election challenges globally.

1. **Follow My Vote (FMV)- Pollaris**

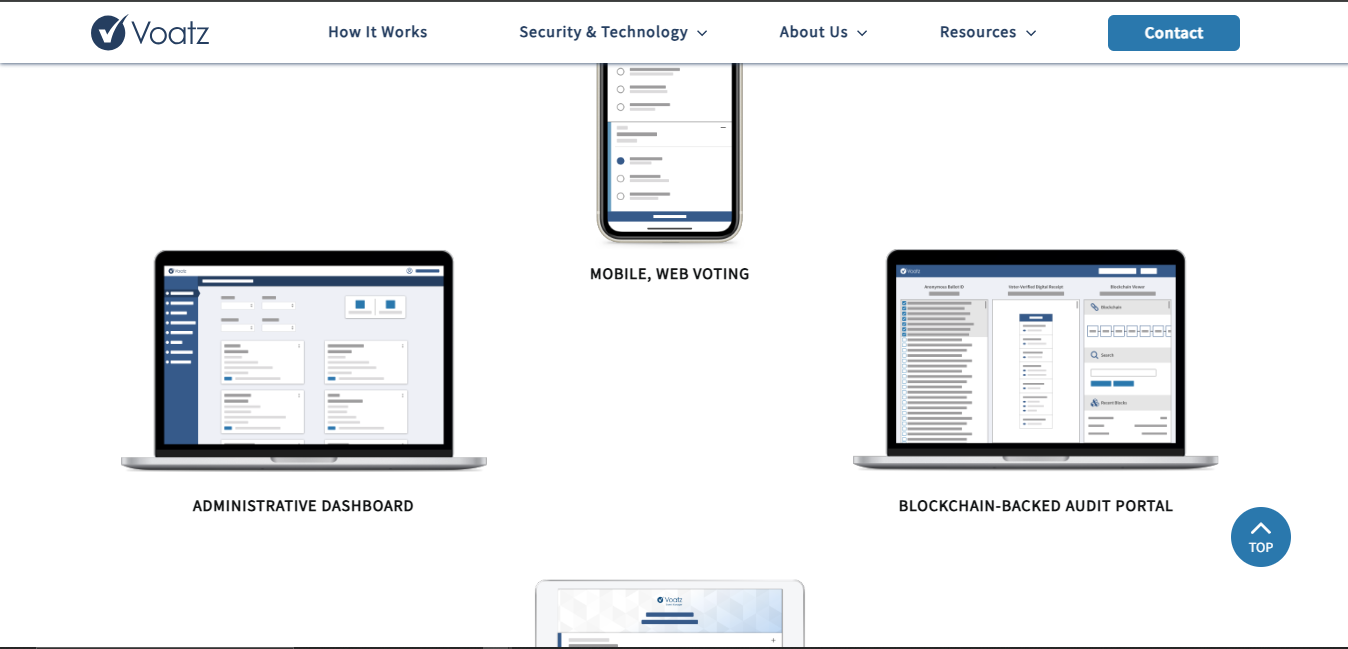
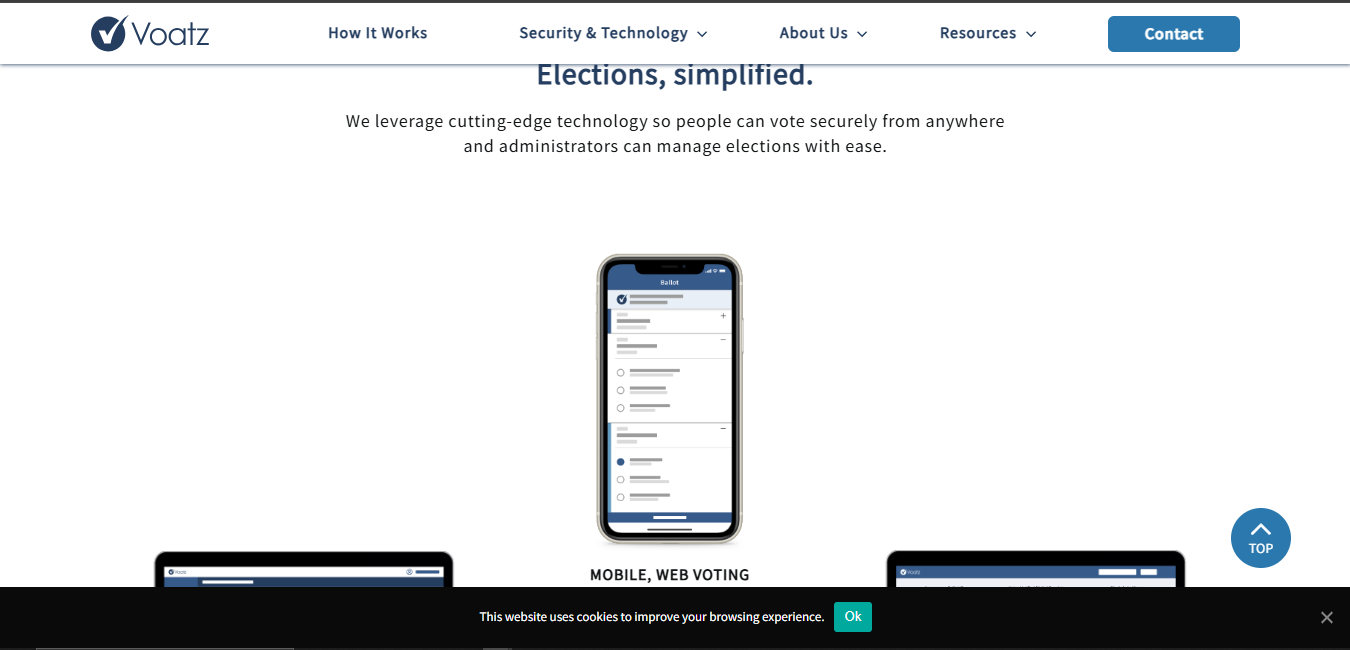
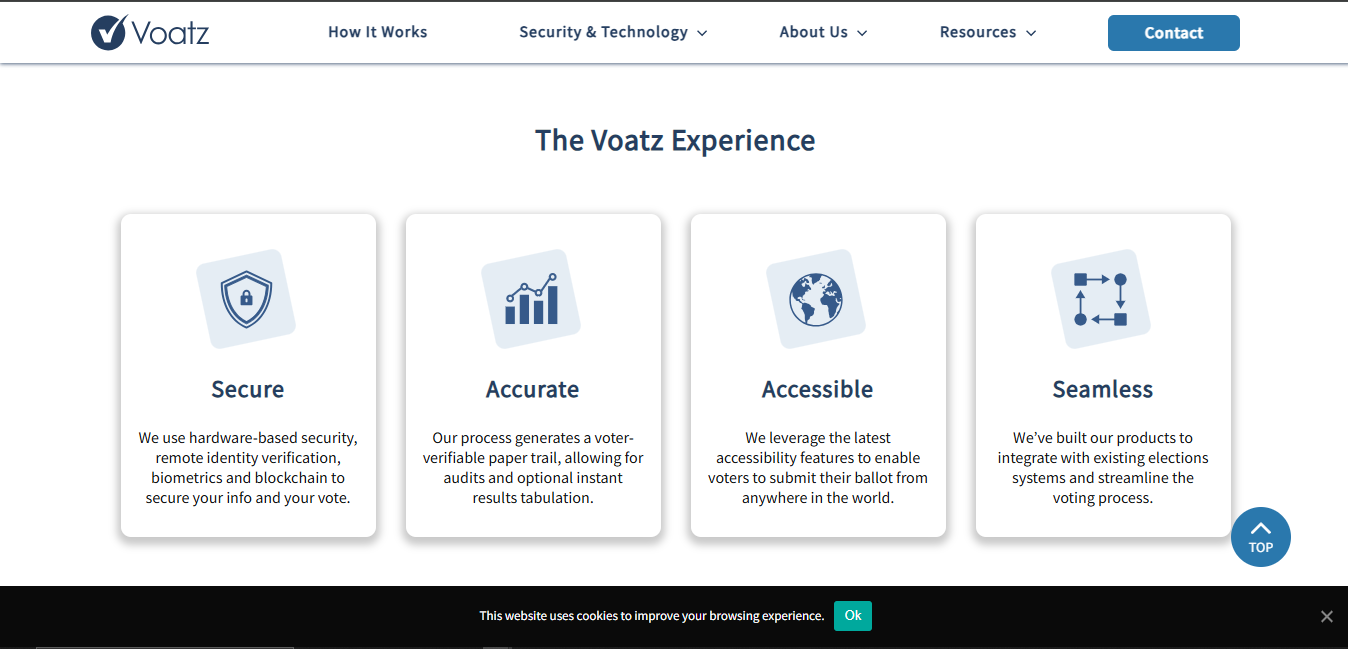
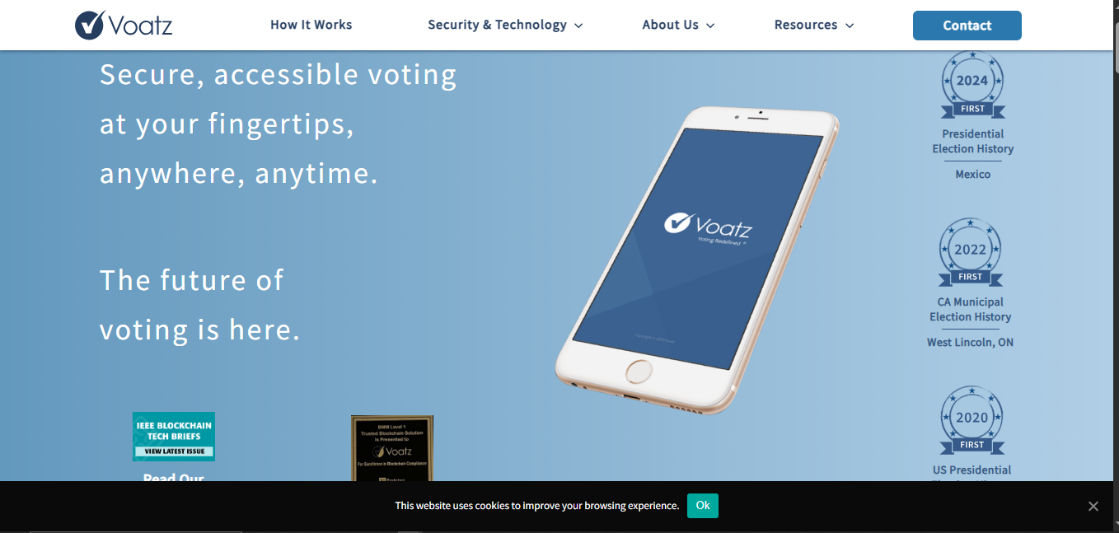
It was designed to offer end-to-end transparency and voter anonymity by recording votes on a public blockchain. However, it has encountered numerous challenges example scalability and high transaction fees.  
Research has shown that public blockchains experience congestion under high transaction load which in return leads to affecting system performance.

Follow my vote- <https://followmyvote.com/>

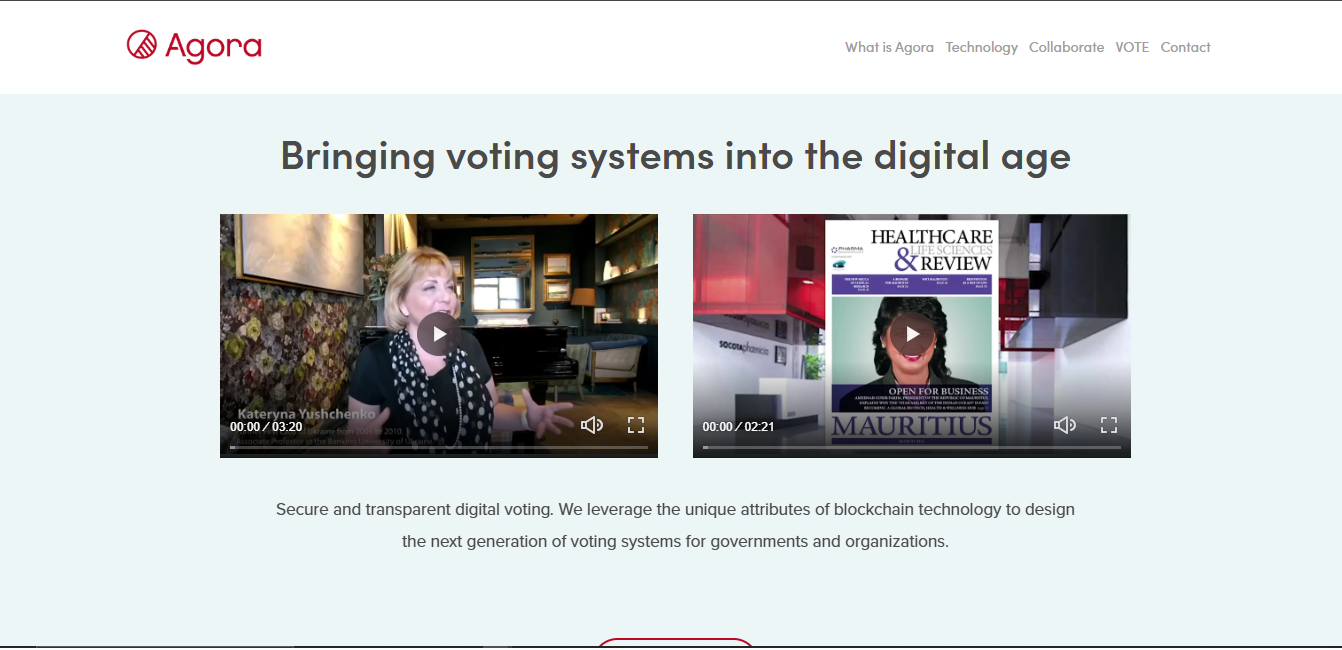
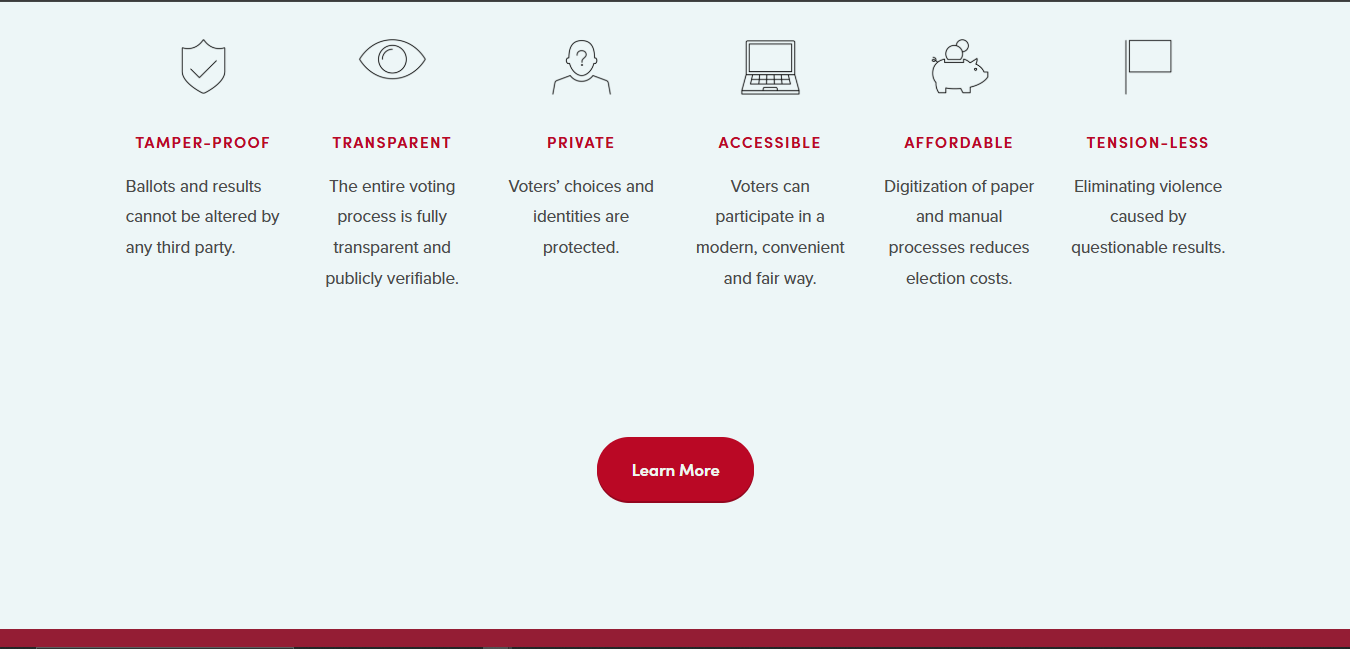
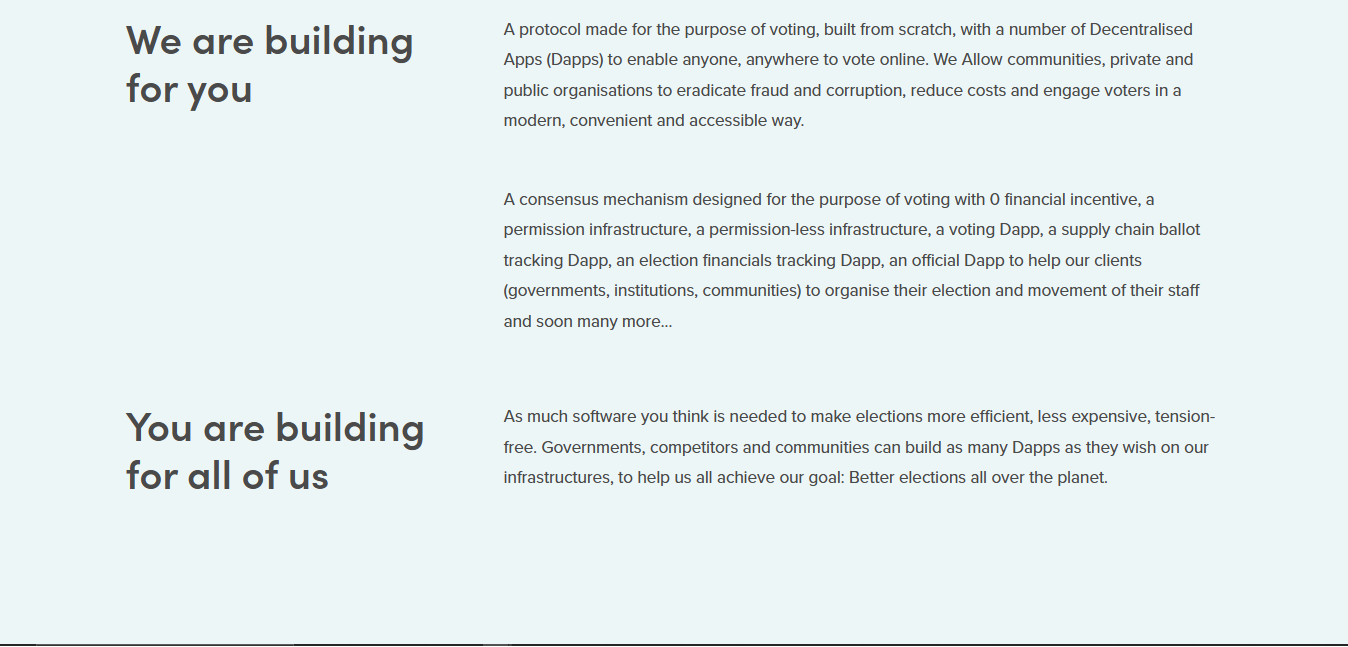




1. **Voatz**  
   It’s a mobile application that leverages biometric authentication to secure the voting process. While this mobile application enhances accessibility, it has faced various challenges e.g. security vulnerabilities and lack of open-source transparency. The proprietary nature of the software it uses has raised questions about independent security audits and trust.

Home- Voatz secure convenient voting anywhere. <https://voatz.com/>

1. **Agora**  
   Agora uses a permissioned blockchain combined with some cryptographic techniques e.g. Zero-knowledge Proofs, that ensures end-to-end verifiability. Although its nature of immutability of votes, its dependency on a single blockchain infrastructure raised concerns about network congestion, single points of failure and transaction processing speed  
   Agora vote <https://www.agora.vote/>

****

## **2.2 Tools and methodologies**

Below is a breakdown of the tools and methodologies used in existing analysed blockchain systems.

|  |  |  |
| --- | --- | --- |
| System | Tools & Technologies applied | Methodologies |
| **Follow My Vote (FMV)** | Ethereum Blockchain, Cryptographic hashing | Voters verify their identity through cryptographic keys and cast their votes on the blockchain. |
| **Voatz** | Mobile application, Biometric authentication, blockchain Ledger | Voters use biometric authentication to register and vote via a mobile application. Votes are stored on a private blockchain |
| **Agora** | Permissioned blockchain, Zero-knowledge Proofs. | Votes are stored on a permissioned blockchain with cryptographic verification mechanisms. |

**Advantages and Disadvantages of the reviewed systems**

1. **Follow My Vote (FMV)**

Pros-high transparency and tamper resistance.  
cons- scalability issues and high transaction fees due to reliance on a public blockchain.

1. **Voatz**pros- increased accessibility and enhances security measures through biometric verification.  
   cons-security vulnerabilities arising from proprietary software and reduced transparency in the absence of an open code.
2. **Agora**pros- provide robust transparency and data integrity.  
   cons- dependence on centralised management and scalability challenges under high transaction volumes.

## **2.3 Gaps in the existing systems and the proposed solution**

1. **Scalability issues**  
   Research shows that public blockchains, while they are secure, they do struggle to process large numbers of transactions simultaneously leading to delays (Johnson et al 2021). This is a serious issue in elections with higher voter turn out.
2. **Transaction costs**the reliance on networks e.g. Ethereum often results in high gas fees making large-scale adoption economically impractical.
3. **Security concerns**As we know that any practical system is vulnerable to attacks. Despite advanced cryptographic methods, vulnerabilities exist particularly in closed or proprietary systems.   
   Also, the lack of transparency hinders third-party verification.
4. **User accessibility**many systems require the users to have a certain level of knowledge in order to use the system. This leads to limiting the adoption among the general public.  
   Usability studies have shown that ease of use is paramount for broad-based voter participation (smith et al, 2022).

## **2.4 The proposed solution**

This study proposes a decentralized voting system using blockchain for small scale organizational elections addressing the identified gaps.

1. **Implementing an off-chain voting process for non-critical Operation:**   
   this will be achieved by handling some processes off-chain e.g. voter registration, which would lead to reducing congestion on the main blockchain thereby enhancing systems performance.
2. **Ensuring a secure authentication module:**the system will use robust cryptographic verification to prevent voter fraud and unauthorized access. This ensures that each vote is tied to a verified voter identity
3. **Enhancing user accessibility:**  
   A user-friendly interface will be developed to simplify the voting process lowering the barrier for non technical users and promoting widespread adoption.
4. **Decentralized vote tallying mechanism:**real-time vote counting and transparent result verification will be achieved through decentralised ledger technology, ensuring the accuracy and integrity of the election is achieved.

# **CHAPTER 3: METHODOLOGY**

This chapter provides a detailed explanation of the methodology used in developing the proposed decentralized voting system. It covers data collection, analysis, implementation, testing, timeline, costs, and the rationale for the chosen methodology, ensuring a structured plan to address the research question: *How can blockchain enhance election integrity in small-scale organizational settings?* The methodology ensures a meaningful, practical experience, culminating in a functional system with scholarly and practical contributions.

## **3.1 Methodology and Tools**

The system will be developed using the **Agile Evolutionary Methodology**, it’s a project management framework that breaks projects down into several dynamic phases. It’s an iterative approach combining incremental development with adaptability. (Sarah L. (2025).

Fig 3.1: phases in agile methodology.  


Reasons for Choosing Agile Evolutionary for my work:

**Flexibility**: Adapts to blockchain learning curves and unexpected challenges.

**Incremental Progress**: Delivers a working prototype iteratively, aligning with the time period of the project

**Practicality**: Suits a proof-of-concept over rigid models like Waterfall, allowing refinement based on testing.

**Focus on User Needs:** Iterative UI development ensures usability, vital for Kenyan organizational voters.

#### Phases:

1. Requirement gathering and analysis: This involves identifying the users and system requirements.
2. System design: doing some system architect and selecting tools for use.

Use of Entity-relationship diagram: map voter, vote and admin data relationships for the off-chain database.  
use of flowcharts to illustrate the workflow.  
use of Data Flow Diagrams to show data movement between frontend, blockchain and off-chain storage.

1. System development: build some smart contracts, modules authentication, UI and tallying.
2. Testing: validate functionality of the system and perform some integration tests.  
   running tests to collect feedback and integrate it with necessary repairs or addition.
3. Deployment and documentation: Finalize my prototype and write and give my report to supervisor.

\*Will have some iterative cycles for continuous refinement. (2 weeks maximum)

## **3.2 Source of Data**

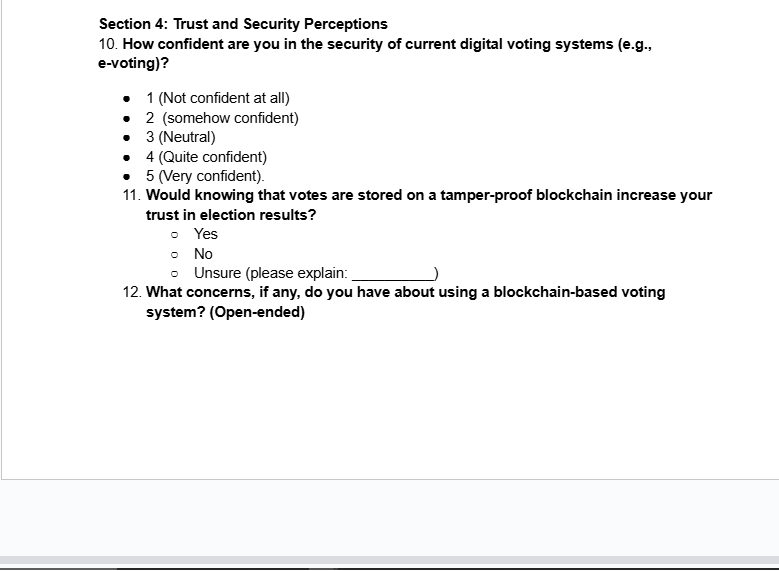
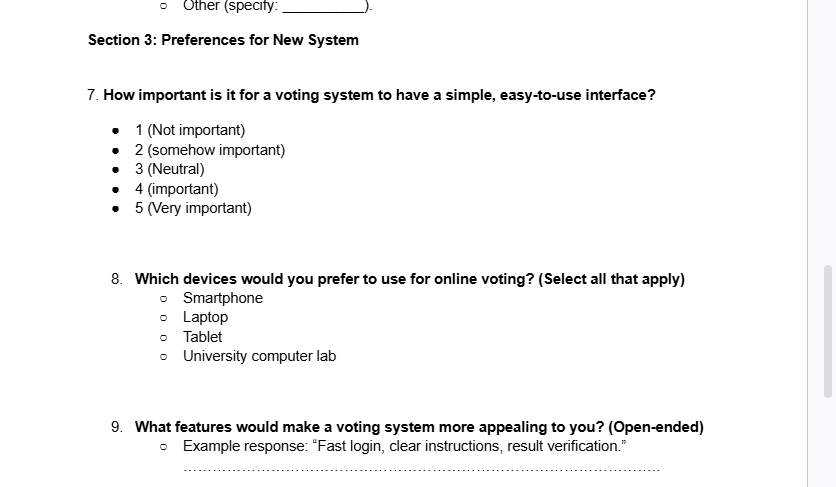
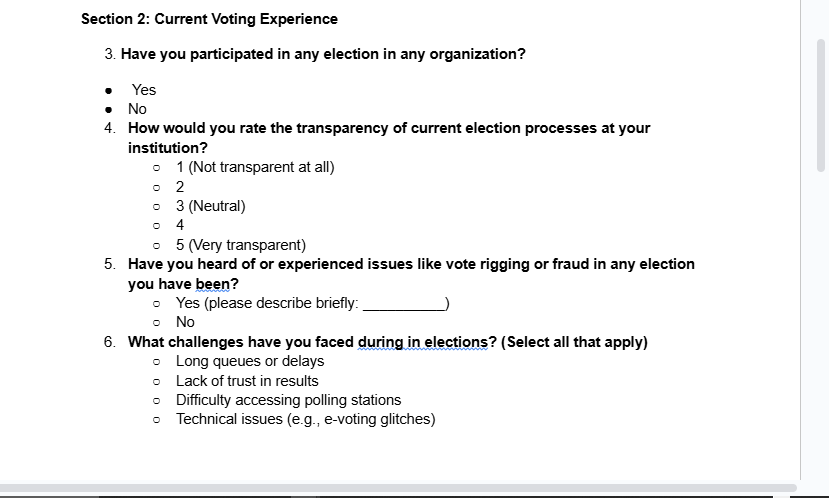
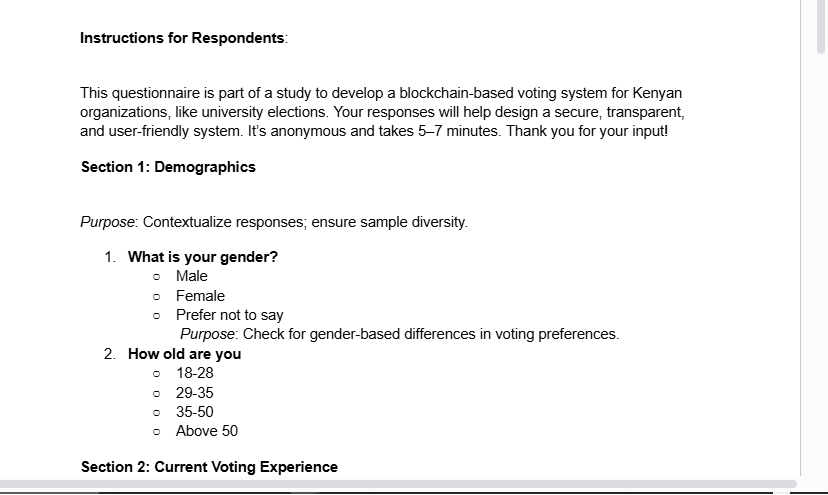
The project requires the primary and secondary data sources as it relies more on a practical and also theoretical solutions.

Primary data  
Primary data refers to information collected firsthand by the researcher for a specific purpose. (Rabianski, 2003)  
feedback from election participants. 10 participants  
observation during system testing. E.g. system errors

Secondary data  
Secondary data refers to data that is not compiled by the analyst, which may include published or unpublished work based on research that relies on primary sources. (Rabianski, 2003)  
Peer reviewed papers on blockchain voting from Google Scholar  
Existing systems documentations.  
Online resources. E.g. online tutorials

## **Data Collection Methods**

Several methods of data collection were used in order to make a well refined decentralized blockchain voting system.

**Questionnaires**Getting more information about what voters feel requires end to end details so using questionnaire was another way of data gathering. This is to assess the user usability and their trust in digital voting platforms.  
Some of the readily available tools were used such as Google forms and open-ended questions.  


**Observation**Doing some prototype simulation as to note the system performance and user interaction.

**Records and documentation**Looked at some several reports and documentation of various organizations e.g. IEBC reports in order to identify fraud patterns and the transparency gap in the current systems.

## **3.4 resources and materials required**

**Hardware resource**

This will include all the hardware requirements of the system to run Efficiently. Listed on the table below:

**Software resources**

It includes the operating system that will be used, the development tools. Blockchain platforms, databases, frontend and antivirus on respective machine.

|  |  |  |
| --- | --- | --- |
| Resources | description | specifications |
| Hardware resources | Laptop, computer | 8GB RAM 256GB ROM modern processor |
| Software resources | Dev tools-vs code blockchain platforms  documentation tools-Ms word code repository | o/s- windows 10 antivirus- Avast antivirus, Hyperledger fabric / Ethereum test net visual studio, react, CSS, HTML, MySQL GitHub. |
| internet | Reliable internet connection | Fast network |
| Human resource | Support and guidance | Peers and supervisor guidance |
| server | Garnache | Free, no external hosting |
| others | Backup and storage | USB drive |

# **CHAPTER 4: SYSTEM ANALYSIS AND REQUIREMENT MODELING**

This chapter analyzes the current voting systems used in Kenyan organizations, which include manual and electronic voting at different organizations and the model requirements for a proposed decentralized web voting systems using blockchain technology. This chapter entails system analysis tools to evaluate existing processes, identified gaps through data collection and defines requirements for the proposed system.

## **Introduction to System Analysis**

System analysis involves examining current voting systems to understand their workflows, limitations and stakeholder needs so as to form the basis for designing a blockchain-based solution. At different organizations they use manual ballots or centralized voting systems which are prone to electoral fraud and inefficiency. In this chapter I’ll use different modelling tools e.g. DFDs, flowcharts, ERDs to map the design of the system and specify requirements for a secure transparent prototype.

## **Objectives of System Analysis**

The system analysis aims to:

To Map the workflow of current voting systems using DFDs, flowchart and UML use cases.

To Identify limitations, inefficiencies and user challenges in the current system.

To define functional (e.g. secure login, verifiable results) and non-functional requirements e.g. user-friendly Ui for the proposed blockchain system.

To evaluate the project’s technical, economic, operational and schedule viability.

To model the proposed system’s architecture and functions.

## **Problem Definition**

The current election systems suffer from low trust due to the lack of transparency as the main issue. This leads to voter fraud (duplicate voting or identity manipulation) as seen in national elections, security risk such as hacking, inefficiencies like delayed results and lack of result verification.  
The proposed system addresses these by leveraging Ethereum blockchain, cryptographic authentication and a React-based UI that will ensure the system is secure, transparent and efficient while conducting elections.

## **Feasibility Study**

A feasibility study evaluates the current and proposed system across four main critical parts i.e. technical, economical, operational and schedule. The proposed system is designed for small-scale elections in organizations using free tools to ensure viability.

**Technical feasibility  
Current system:** manual ballots require paper and physical polling stations. E-voting systems uses centralized servers which are prone to crashes.  
**Proposed system:** The proposed system will use Ganache (Ethereum test net) for blockchain, MySQL (XAMPP) for off-chain data, React/Node.js for web interface.  
These tools are free and I have interacted with them through my academic journey and are open source and locally deployable.

**Economic Feasibility  
Current system:** Manual voting requires more resources since the papers need to be bought and printed per election; e-voting requires server maintenance which also require more money.  
**Proposed system:** Will use open-source software to minimize the cost. As a student-led project, it can run on local test environments hence no significant capital expenditures needed beyond internet access.

**Operation feasibility  
Current system:** manual voting is labor intensive, e-voting requires IT staff for system management and deterring non-technical users.  
**proposed system:** It’ll have a user-friendly React UI and mobile responsive design suiting the voters. Training the voters will take 10-15 minutes which is feasible for the voters in general.

**Schedule feasibility**The system will be developed over a 12–14-week period aligning with my project timeline. Using Agile sprints ensures timely delivery.

## **System Analysis Tools**

The proposed system uses Agile-compatible modelling tools to design and analyse processes, data, and interfaces, ensuring alignment with Agile Evolutionary methodology. To understand and design the system’s structure and behavior, the following modelling tools are applied:

**Data Flow Diagrams (DFDs):** illustrate data movement between voters, React Ui, Node.js Ganache, and MySQL.  
**Flowcharts:** For modelling voter registration, vote casting and result verification for current and proposed systems. Created in Lucid chart.  
**UML Use Case Diagram**: for illustrating actor interactions (voter, admin) with system functions.  
**Entity-Relationship Diagrams (ERDs): To d**efine MySQL database schema for voters, votes, and elections

**Wireframes:** To design user interfaces (Login, voting results) using Figma.

These tools implemented using free platforms ensure clear visualizations and iterative refinement, supporting Agile sprints.

## **System Investigation**

### Introduction

This section gathers data on the current voting systema i.e. manual ballots, centralized voting system and its shortcomings through both primary and secondary sources so as in order to inform the proposed system’s design.

### 4.6.2 Data Collection Methods

**Questionnaires:** Did some questionnaire through some google forms which were distributed through online forms. The main aim was usability and trust to rate transparency of current elections.  
**Observation:** I attended a planned organization election which was manual voting. I noted some several factors like queuing time and tally delays.  
**Interviews:** engaged with voters, IT staff to get more of the process, pain points and their expectations when it comes to elections. Asked some critical questions e.g. “Have you ever witnessed election rigging?”

### Fact Recording

Facts were recorded in various form using forms, narratives and diagrams in order to define requirements of the proposed system.

**Forms:**

* **Requirement Form (Table 4.1):**

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement** | **Description** | **Current System** | **Proposed System** |
| Input | Results | Manual forms, web input | Web UI (React) with OTP |
| output | Results | Paper reports, web display | Blockchain-verified results |
| Process | Registration, voting, verification | Manual checks, server-based | Node.js, Ganache blockchain |
| File | Voter records, votes | Paper, SQL database | MySQL, blockchain hashes |
| System | Scalability | Limited | Scalable |
| Personnel | Admins, voters | 5–10 staff | |  | | --- | | 1 admin, voter self-service | |

**Narratives:** documented in interviews.

**Diagrams:** system flowcharts, ERD and DFDs which details the activities.

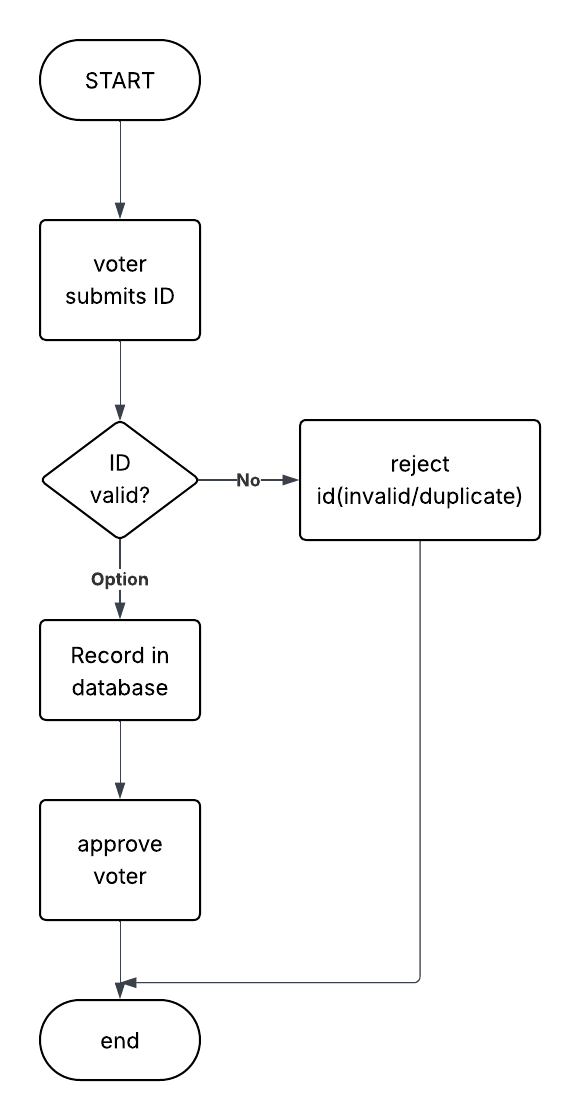
## **System Analysis**

System analysis compared current and proposed systems to define requirements and model processes, using forms, flowcharts, DFDs, and ERDs.

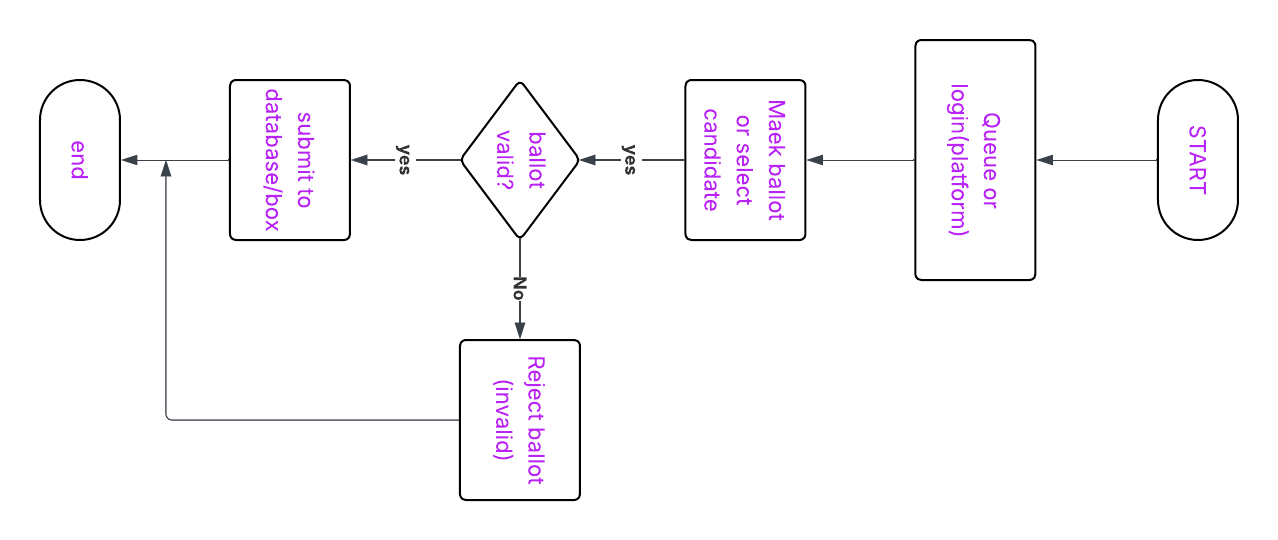
### 4.7.1 Current System Analysis

* **Description:** 
  + Manual voting systems: Voters register with ID cards, mark paper ballots, and results are tallied manually which is time consuming and also, they may be errors and tampering risks.
  + Electronic voting systems: The voters log in via web, select candidates, and results are displayed on the dashboard. This method has less to no transparency and is at risk of hacking.
* **Modelling:** 
  + **Flowcharts:**

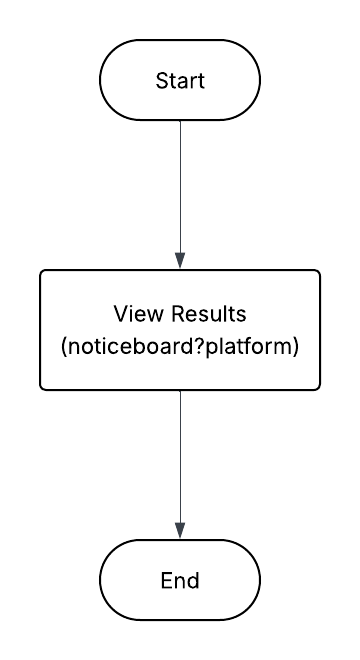
Current registration flowchart



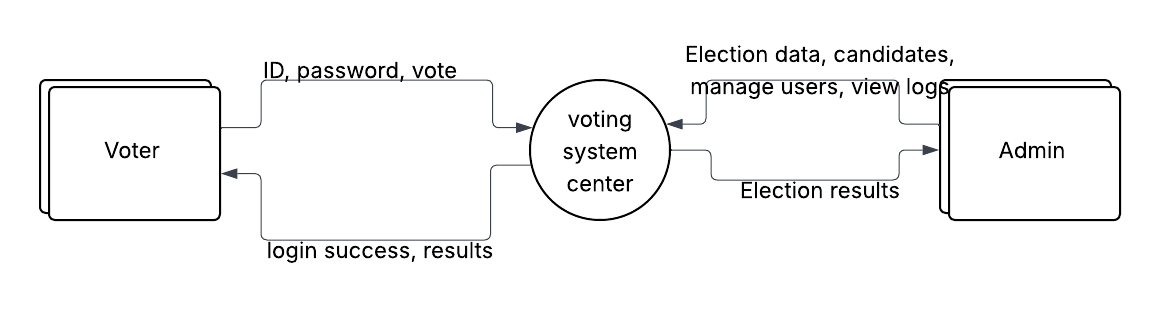
Current voter casting flowchart



Current system result verification



**DFD:**

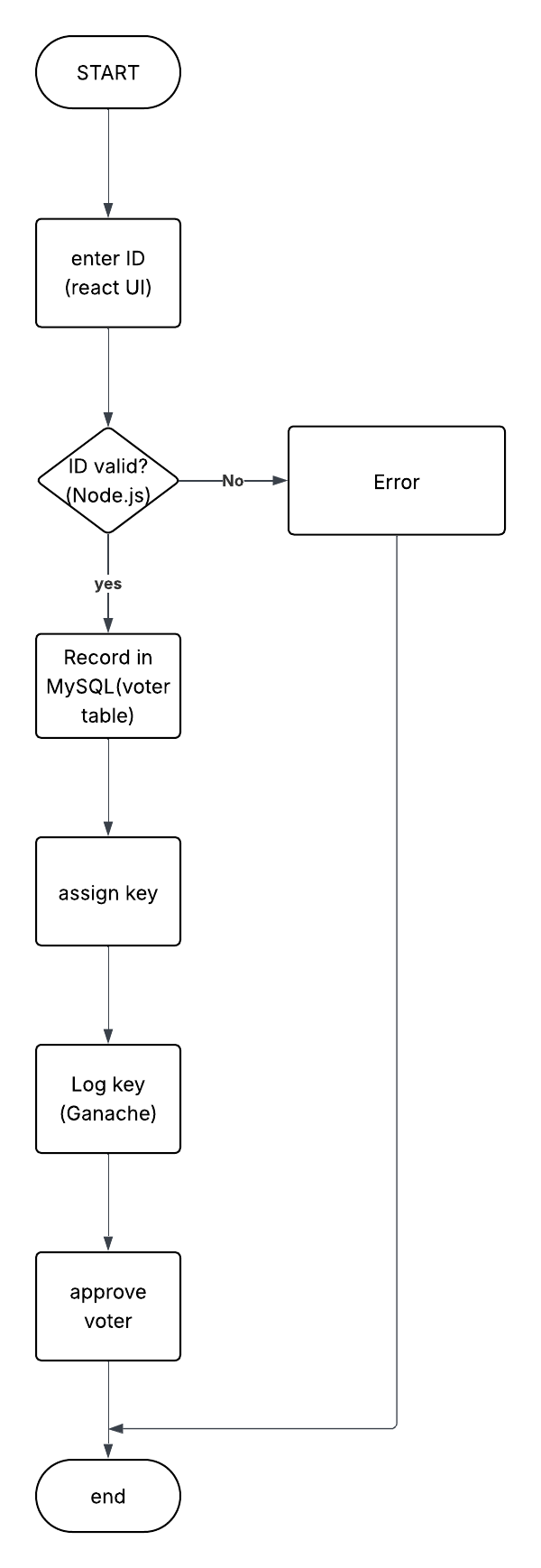


* **Requirements:** 
  + **Input:** Physical ID, ballot mark (manual); username, vote (electronic).
  + **Output:** Paper reports (manual); web results (electronic).
  + **Process:** Slow verification, no audit trail.
  + **Issues:** Reusable requirements (ID input, vote output) but need security, transparency.

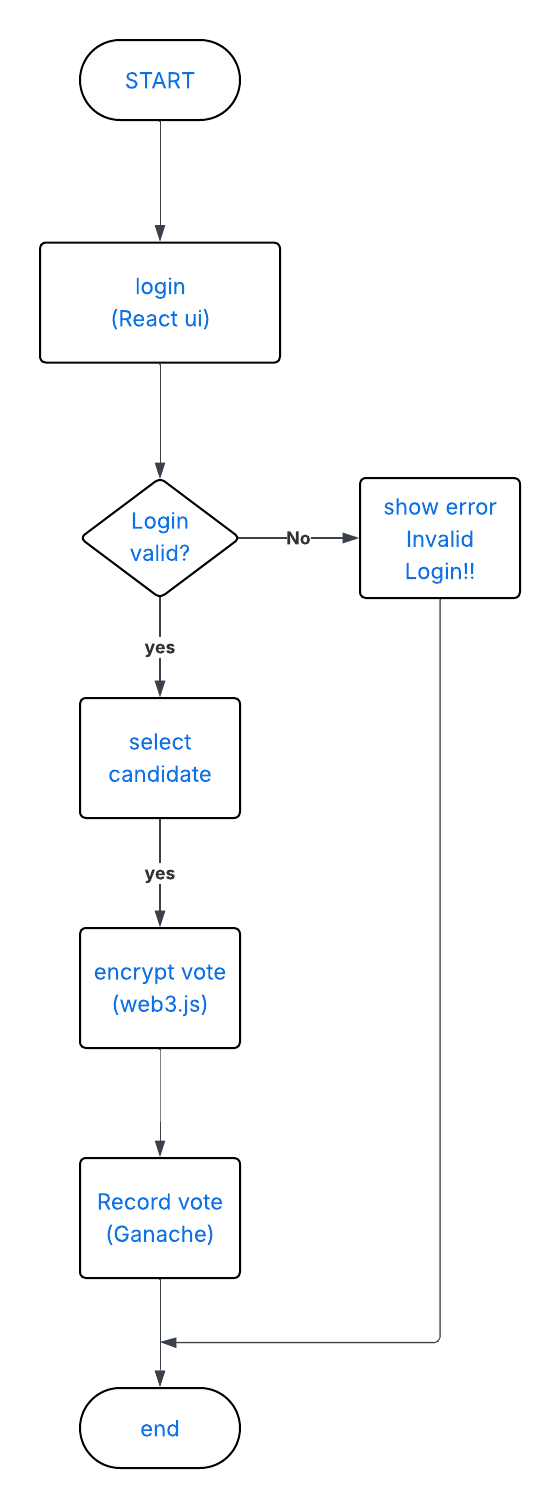
### 4.7.2 Proposed System Analysis

* **Description:** 
  + A web-based system using React (UI), Node.js (backend), Ganache (blockchain), and MySQL (database) for secure, transparent voting.
  + Processes: Voter registers with ID/OTP, casts vote (blockchain hash), verifies results via blockchain explorer.
* **Modelling:** 
  + **Flowcharts:**

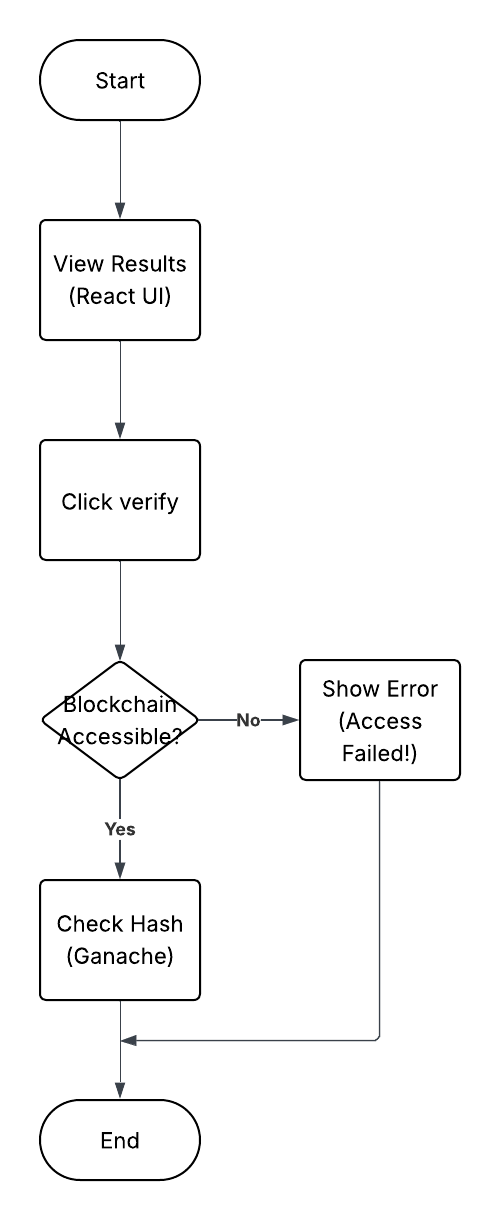
Blockchain registration



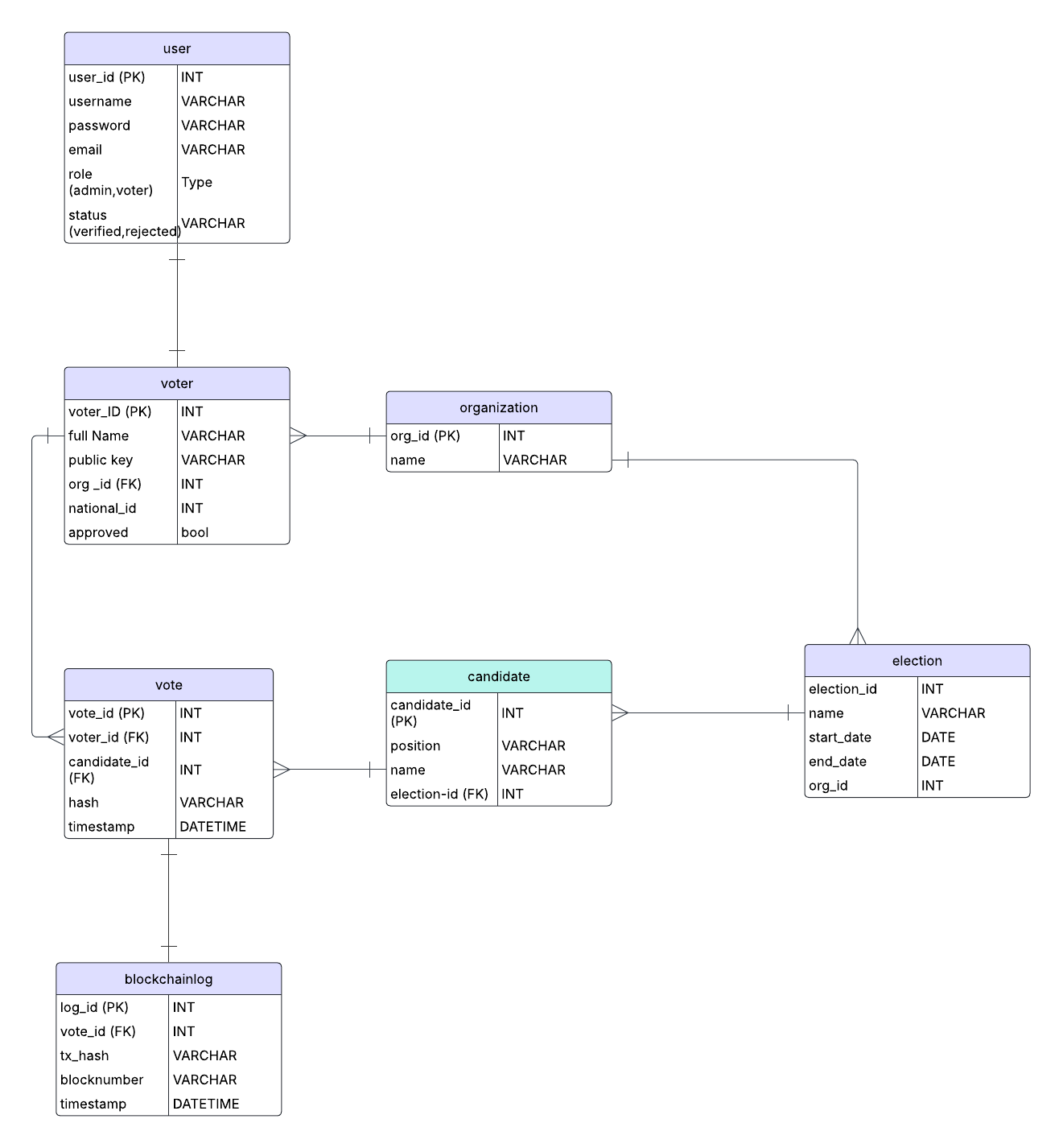
Blockchain vote casting



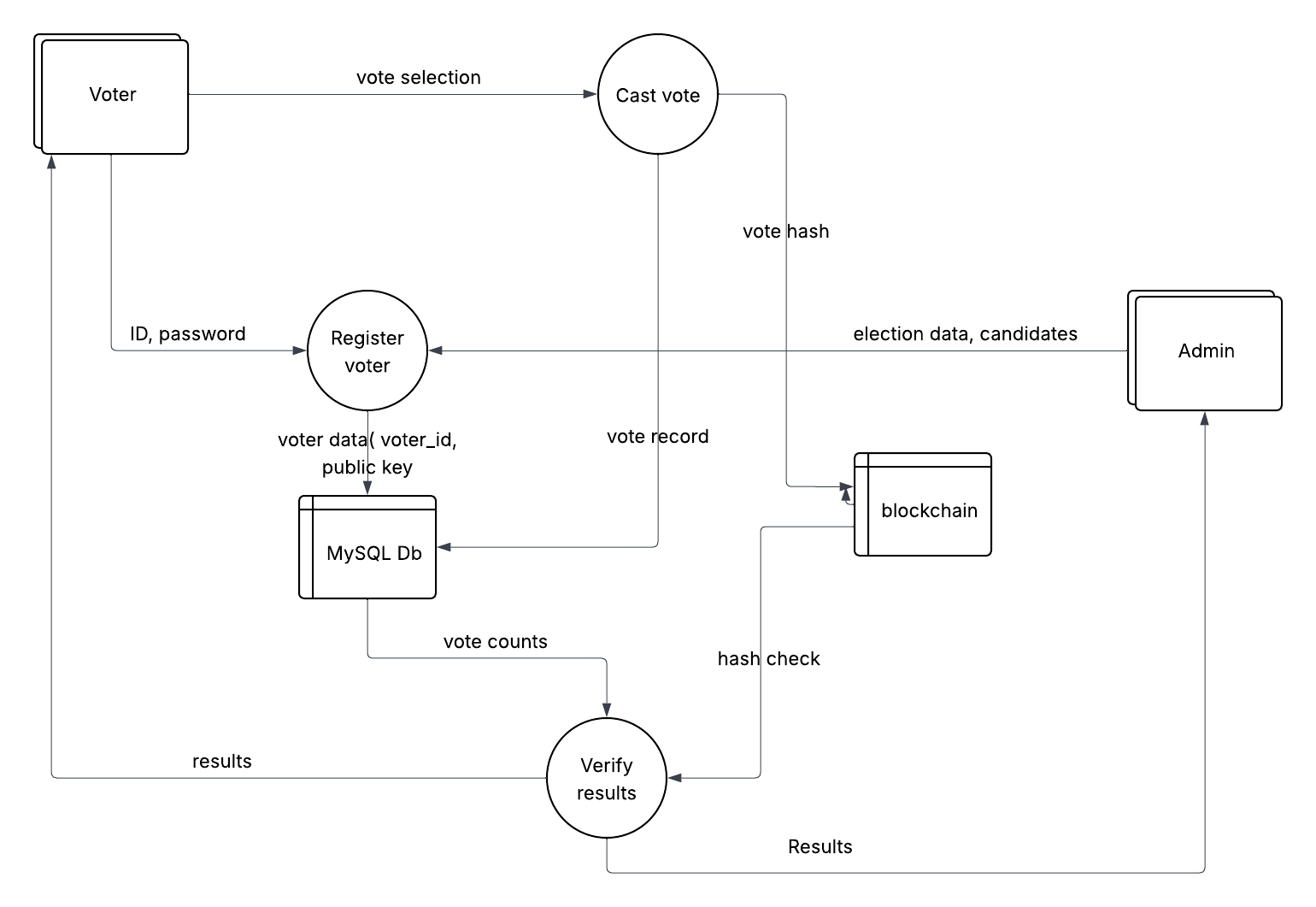
**Blockchain result verification**



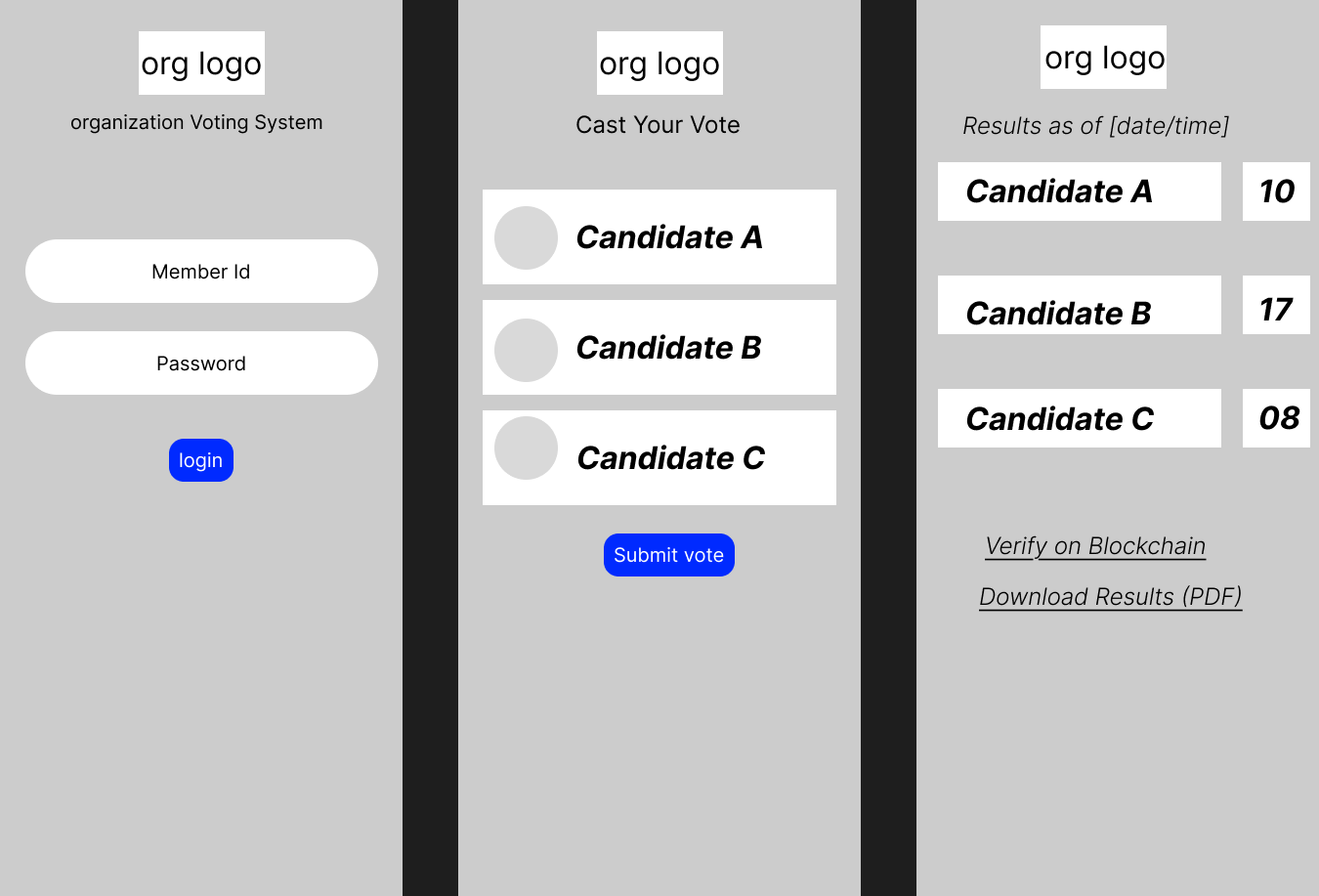
* + **ERD:**



* + **DFD:**



* + **Wireframes:**



* **Requirements:** 
  + **Input:** ID, password, OTP, vote selection.
  + **Output:** Login success, vote confirmation, verifiable results.
  + **Process:** Secure registration (Node.js), tamper-proof voting (Ganache), transparent verification.
  + **File:** MySQL tables (voters, votes), blockchain ledger.
  + **System:** Scalable for organizational usage, user-friendly (React UI).
  + **Personnel:** 1 admin (election setup), self-service voters.

### 4.7.3 Requirement Definitions and Specifications

* **Current System:**

**Functional:** ID-based registration, vote casting, result display.

**Non-Functional:** Limited scalability, low security.

* **Proposed System:**

**Functional:**

* + - Secure login with OTP (Objective 2).
    - Vote recording with blockchain hash (Objective 3).
    - Result verification via blockchain (Objective 4).

**Non-Functional:**

* + - User-friendly UI.
    - Scalability for organizational use
    - Cost-effective.

**Specifications:**

* + - **Database:** MySQL.
    - UI: React, mobile-first.
    - Blockchain: Ganache for hash storage.
    - Security: Public/private keys, OTP.

### 4.7.4 Comparison

* **Current System:** Reusable inputs (ID, vote) but lacks security, transparency, and scalability.
* **Proposed System:** Builds on current inputs, adds blockchain (tamper-proof), OTP (secure), and web UI (accessible), addressing all limitations.
* **Outcome:** Current system requirements (ID input, vote output) are integrated, with enhanced security and transparency for the proposed system.

# **CHAPTER 5: SYSTEM DESIGN**

## **5.1 Introduction to System Design and Nature of the System**

System design translates the requirements from Chapter 4 into a technical blueprint for a secure and transparent decentralized web voting system for Kenyan organizations, including both unions, and saccos. The system is a web-based application integrating a React front-end, Node.js back-end, Ganache blockchain for vote integrity, and MySQL database (via XAMPP) for voter and election data. It supports organizational voting culture, ensuring user-friendly interfaces, secure authentication, tamper-proof voting, and verifiable results. The design is modular, scalable, and cost-effective using an Agile methodology to fully develop the system. This chapter covers conceptual, logical, and physical designs, including database models, input/output screens, and program flowcharts, with deliverables. It’s a web-based system where voters log in, in, cast votes, vote, and verify results, and admins manage elections, all while keeping everything safe and trustworthy.

**Nature of the system:  
Decentralized:** Uses blockchain (Ganache) to ensure votes can’t be tampered with, building trust for the organization personnel.

**Mobile friendly:** React UI works on different digital devices including smartphones hence it is readily accessible to members of organizations.

**Scalable:** can handle small scale elections to larger ones.

**Secure:** Protects voters’ data with OTP authentication and blockchain hashes.

This chapter will cover:

**Database design**. i.e. who’s, who and what they are stored.

**UI screens**- how people interact with the system i.e. both voters and admins.

**The program flow**- how the system works step by step.

**Security and performance of the system.**

## **Objectives of System Design**

The main objectives of the project include:

* Build a clear plan: Designing a system with the use of different tools and programs like React, Node.js, Ganache and MySQL that will be easy to understand and build.
* Organize data: Creating a database to store voters, votes and election data securely and efficiently.
* Make it user-friendly-: Designing screens for Loging, voting, results that are easy to follow and understand for even non-tech users.
* Security: Ensuring maximum security by using blockchain and OTP to protect votes and voter info in order to boost trust.
* Plan for growth: make the system scalable by working for small scale organizations to large scale organizations.
* Documentation: BY using of diagrams and code plans to guide the project development.

## **Program Design Tools**

With the use of Agile methodology approach, I’ll design the system with the use of free tools which include the following:

* Entity-Relationship Diagram (ERD): To show how data (voters, votes) is organized in MySQL.
* Data Flow Diagrams DFDs): It maps how data will move from voter to blockchain and vice versa.
* Flowcharts: Draws step by step processes for the whole systems and individual parts.
* Wireframes: Sketches of UI screens in Figma.
* Data dictionary: It will list all database details like a user manual. Lists and describes all data elements, their types, and usage.
* Algorithms (Pseudocode): Describes logic of processes in plain English before coding i.e. write code-like instructions for key tasks like voting.
* UML Use Case Diagrams: For modelling object-oriented aspects where applicable.

## **Logical Design**

Logical design involves organizing data by setting how it is structured in the database for efficiency.

### Logical Data Design: Relational Data Analysis / Normalization

Normalization is the process of organizing data to avoid duplication. With the use of ERD from the previous chapter, it’ll assist in refining the data into tables using 1NF, 2NF and 3NF.

* **1NF (First Normal Form):** making sure each field has one value and a unique ID. E.g. splitting “voter name” to “first name” and “last name” for easier search.
* **2NF (Second Normal Form):** ensuring that all fields depend on table’s ID. E.g. moving candidates’ details to a separate candidate table linked by candidate\_id.
* **3NF (Third Normal Form):** Removing fields that depends on other fields, not the ID. E.g. moving organization table, linked by org\_id.

Normalized Tables:

1. Voter: stores voter information (ID, name, blockchain key)
2. Candidate: lists candidates’ info (name, position)
3. Vote: record votes with blockchain hash.
4. Election: defines election events (name, dates).
5. Organizations: represents organizations (unions, saccos, universities)
6. Blockchain Log: log\_id, action, hash, timestamp.

### Entity Attributes Relationships

The ERD shows how tables connect in the system. They include:

**Voter → Organization:** Many voters belong to one org (e.g., Safaricom).

**Voter → Vote:** One voter can cast multiple votes (across elections).

**Candidate → Election:** Many candidates run in one election.

**Vote → Candidate:** Many votes go to one candidate.

**Election → Organization:** Many elections are held by one org.

Table relationships:

|  |  |  |  |
| --- | --- | --- | --- |
| TABLE | LINKED TO | RELATIONSHIP | KEY |
| Voter | Organization | Many-to-One | Org\_id (FK) |
| Voter | vote | One-to-Many | Voter\_id (FK) |
| Candidate | Candidate | Many-to-one | Election\_id (FK) |
| Vote | Candidate | Many-to-one | Candidate\_id (FK) |
| Election | Organization | Many-to-one | Org\_id (FK) |

### Entity Life History

The ELH will describe the lifecycle of key entities involved in the system. ELH aids in tracking data change overtime and also aid in data retention. Below is a breakdown of the Entity Life History for the main entities, explaining their lifecycle stages from creation to deletion (if applicable), and what triggers each change:

* **Voter ELH**

|  |  |  |
| --- | --- | --- |
| **Stage** | **Event cause** | **Description of the event** |
| Created | The system admin registers a new voter. | The new voter profile is inserted to the existing voter table with a public key. |
| Updated | The voter updates their profile information with authentication required. | The Voter updates their email, phone number or password. |
| Active | The voter logged in successfully with an OTP validation | The voter is allowed to vote after OTP verification. |
| Voted | Vote cast successfully. | The Voter\_id is stored in vote table, flagged as voted. |
| Deactivated | Elections ends/voter removed | The Voter is marked inactive or removed by the admin. |

**Candidate ELH**

|  |  |  |
| --- | --- | --- |
| **Stage** | **Event cause** | **Description of the event** |
| Created | The system admin adds a candidate. | The candidate’s info is inserted into candidate table. |
| Active | The election is ongoing. | The candidate is eligible to receive votes. |
| Updated | Admin edits candidate details. | The candidate info is modified e.g. position or age. |
| Archived | The election ends. | Results are stored and the candidate’s info is no longer modifiable |

**Election ELH**

|  |  |  |
| --- | --- | --- |
| **Stage** | **Event cause** | **Description of the event** |
| Created | The admin creates new election | The election is added to election table with metadata (start and end, dates) |
| Started | The election start date is reached | Voting is enabled for all eligible voters registered in the system. |
| Ongoing | The election is taking place | The system accepts votes and tallies in real time i.e. voters cast their votes |
| Ended | The election end date and time are reached. | The voting stops and results are viewable |
| Archived | The system admin archives the election | Elections are locked for viewing only with no edits allowed. |

**Vote ELH**

|  |  |  |
| --- | --- | --- |
| **stage** | **Event cause** | **Description of the event** |
| Created | The voter casts a vote | The vote is inserted into vote table |
| Confirmed | Blockchain transaction complete | Vote hash stored from Ganache blockchain. |
| Counted | The vote is included in tally | Tally is updated per candidate |
| verified | Admin audits vote integrity | Blockchain is validated for authenticity. |

**Organization ELH**

|  |  |  |
| --- | --- | --- |
| **Stage** | **Event cause** | **description** |
| Created | New organization gets a version of the system for their own organization (admin level) | Added to organization table by the system admin |
| Linked | Voters and elections are created. | Voter and election records reference org\_id. |
| Updated | The system admin updates organization details. | The system admin updates basic info e.g. name, description changes and not the organization data. |
| Retired | Organization no longer active. | Set as inactive, system prevents future elections. |

**Blockchain ELH**

|  |  |  |
| --- | --- | --- |
| **Stage** | **Event cause** | **Description** |
| Created | Any major action e.g. login, vote etc. | Event logged with timestamp and hash. |
| Queried | Admin or system reads for audit. | Read-only access to blockchain activity |
| Permanent | Blockchain logs are immutable | No edits or deletions allowed! |

**Summary table of the system**

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity** | **Creation event** | **Main event** | **end/inactive events** |
| Voter | The admin adds voter | Login. Casting vote, update info. | Removed or the election ends |
| Candidate | The admin adds candidate. | Receive votes from the voters | Election ends or removed. |
| Election | Admin creates election | Voting starts | Election ends or archived. |
| Vote | Voter submits vote | Blockchain confirmed, tallied | Read-only record forever |
| Organization | Admin registers an organization. | Manages their elections and voters. | Deactivated by admin. |
| Blockchain-log | As system event occurs | Read-only blockchain tracking | immutable |

## **Physical design: Building the System**

Physical design of the system involves the real-world setting up of the databases, screens and codes. It’ll turn the logical plan into real components: a MySQL database to store data, React screen for users to interact with, node.js code to process data and Ganache blockchain to secure votes. This section creates the database, designs inputs/output screens, sets up unique keys and maps how the systems parts will work together.

### 5.5.1 Data dictionary

The data dictionary defines physical attributes for MySQL tables; it explains every field in MySQL database. It defines what each table stores, its format and rules (e.g. this field can’t be empty). The table below describes the data dictionary of this system

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Field** | **Type** | **Size** | **Constraints** | **Description** | **Example** |
| **Voter** | voter\_id | INT | 8 | PK, Auto-increment | Unique voter ID number | 40401010 |
|  | first\_name | VARCHAR | 15 | Not Null | voter’s first name | steve |
|  | last\_name | VARCHAR | 15 | Not Null | voter’s last name | Otieno |
|  | public\_key | VARCHAR | 255 | Not Null | Blockchain public key | 0x123abc... |
|  | org\_id | INT | 11 | FK (Organization.org\_id) | Links to org | 1 (Smart cycle org) |
| **Candidate** | candidate\_id | INT | 8 | PK, Auto-increment | Unique candidate ID | 20102010 |
|  | first\_name | VARCHAR | 15 | Not Null | Candidate’s first name | Janet |
|  | last\_name | VARCHAR | 15 | Not Null | Candidate’s last name | Wambui |
|  | position | VARCHAR | 30 | Not Null | Role (e.g. manager) | manager |
|  | election\_id | INT | 11 | FK (Election.election\_id) | Links to election | 301 |
| **Vote** | vote\_id | INT | 11 | PK, Auto-increment | Unique vote ID | 401 |
|  | voter\_id | INT | 11 | FK (Voter.voter\_id) | Links to voter | 101 |
|  | candidate\_id | INT | 11 | FK (Candidate.candidate\_id) | Links to candidate | 201 |
|  | hash | VARCHAR | 64 | Not Null | Blockchain vote hash | 0xabc123... |
|  | timestamp | DATETIME | - | Not Null | Vote time | 07/05/2025 10:00 |
| **Election** | election\_id | INT | 11 | PK, Auto-increment | Unique election ID | 301 |
|  | name | VARCHAR | 100 | Not Null | Election name | samrt cycle org 2025 Election |
|  | start\_date | DATE | - | Not Null | Start date | 01/05/2025 |
|  | end\_date | DATE | - | Not Null | End date | 03/05/2025 |
|  | org\_id | INT | 11 | FK (Organization.org\_id) | Links to org | 1 (smart cycle org) |
| **Organization** | org\_id | INT | 11 | PK, Auto-increment | Unique org ID | 1 |
|  | name | VARCHAR | 100 | Not Null | Org name | smart cycle org |

### 5.5.2 File/Database Design: Setting Up MySQL

This is where the building of the actual database in XAMPP(MySQL) to store voter, election and vote data happens.   
database name: voting\_system.

The data won’t be typed into MySQL directly but by use of node.js which will handle data insertions when users submit forms. E.g. voter registration – “insert into voter...”

Example: when a voter registers Node.js runs:

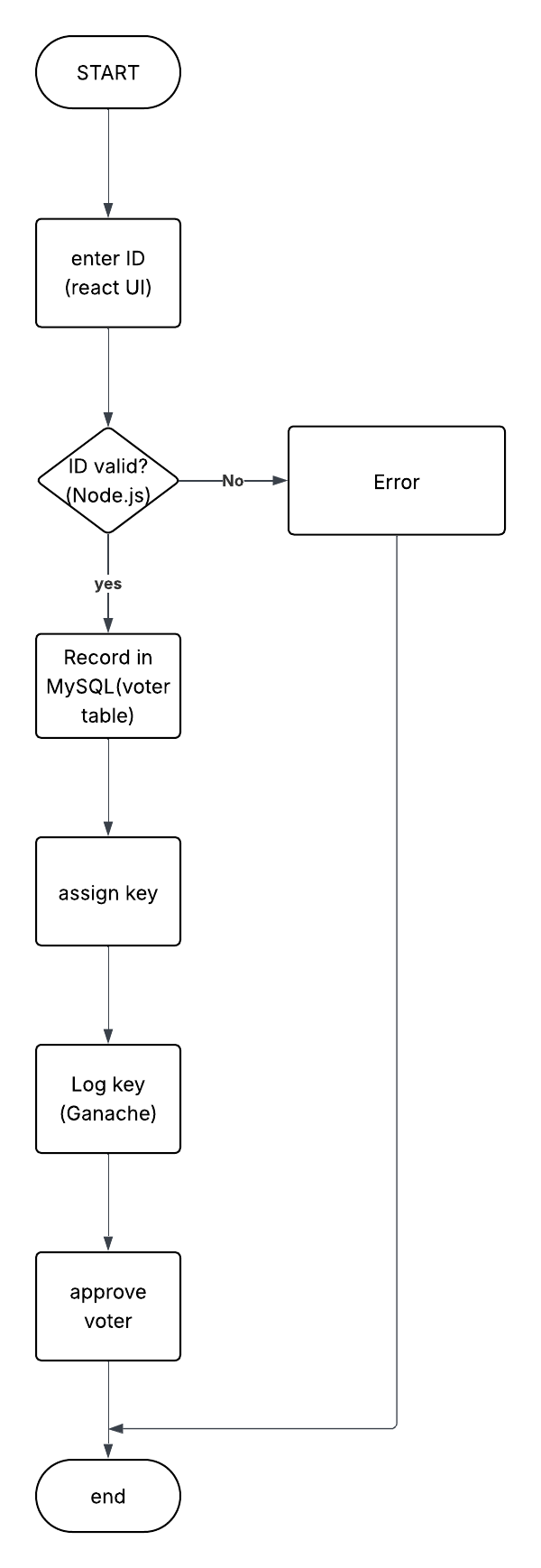
“INSERT INTO voter ((first\_name, last\_name, public\_key, org\_id) “

“VALUES ('Mark', 'Njoroge', '0x123abc...', 1);”

### 5.5.3 Input Screen Design: How Users Enter Data

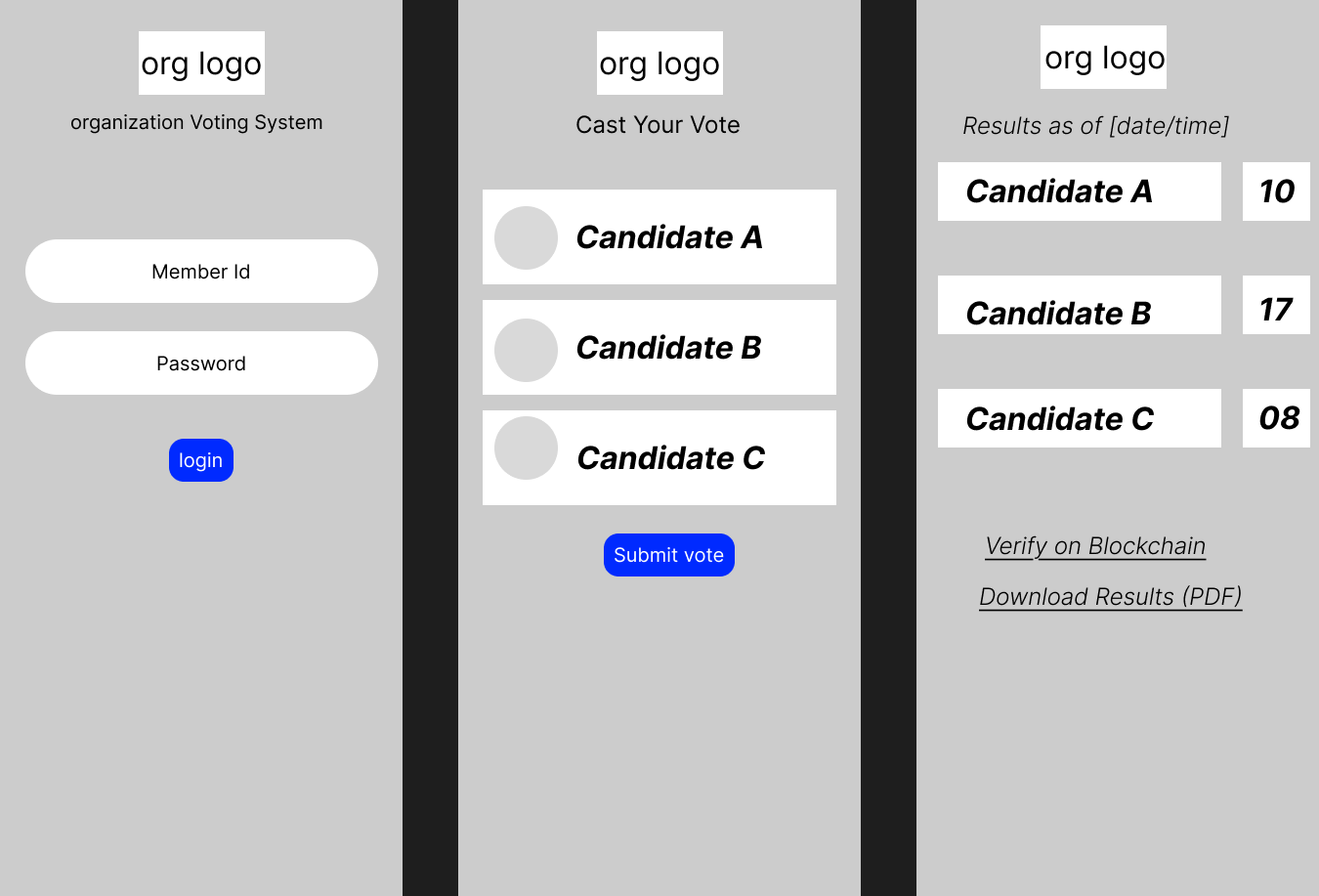
Input screens are the web forms which allows users to enter their data into the system. This is how the data gets into the system and not by manual typing into MySQL is required.

**1. Voter registration**

****

The flowchart directs that the voter actions like inserting their respective names, numbers etc. This action leads to sending data to node.js which inserts them into the voter table. The required fields include member id, first and last name, password and OTP checkbox.

**2. vote casting**



On the cast vote screen, the candidates are in a list format. choosing one person just for one chance is the only possible move hence submitting your vote will send choice to node.js which in turn leads to ganache generating hash and the data is stored in vote table.

**3. Election setup**

Election setup involves the system admin setting up a new election for their respective organisation. The fields involved in this process include the election name, the start and end date (date pickers), and the candidate involved in the election.

**4.organization setup**

It involves setting up new organizations to the system.

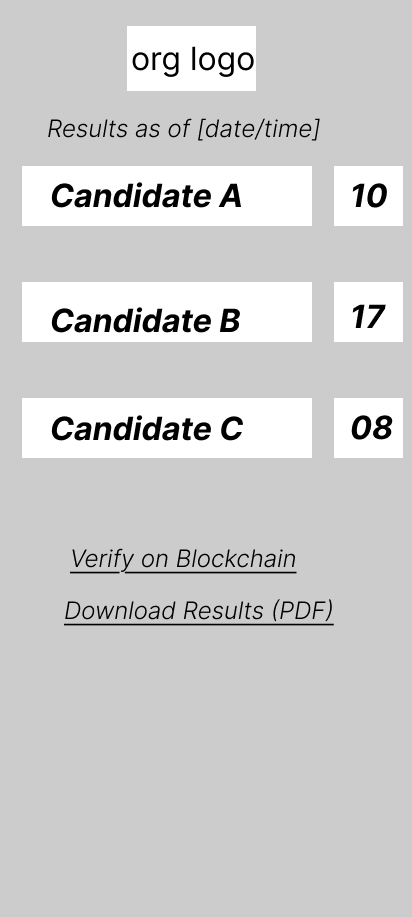
## So how does the data flow in the system?

* Users fill form on react app. i.e. first name, second name, member id
* React send data to node.js via API e.g. POST/register.
* Node.js validates the data, runs SQL to insert into MySQL or hash to ganache.

### 5.5.4 Output Screen Design: Showing Results to Users

The output screen displays data input via the web.

Results report



The screen displays the candidates name, the total number of votes and position they are vying for.

Election summary

The screen shows the election details like name, dates, turnout and results link

Voter list

The voter list screen consists of voters’ data i.e. names, id and is only viewable to admin.

### 5.5.5 code design: unique keys for the system

Keys are necessary and crucial when it comes to data management. The keys ensure every record is unique and properly linked e.g. ID numbers for people. Keys used in the system include:

* **Primary keys –** used for Unique Ids for each table.

A list where primary keys are crucial include: voter\_id, candidate\_id, vote\_id, election\_id, org\_id.

INT specifies that the data type of the column is an integer

auto increment function instructs the database to automatically generate a unique, sequential number for this column whenever a new row is inserted into the table.

e.g voter\_id=101 for Mark Njoroge.

* **Foreign keys:** Used for liking tables.

e.g. vote.voter\_id → voter.voter\_id ensures that votes are coming from valid voters.

* **Blockchain hash:** VARCHAR(64), is a unique vote identifier from ganache.

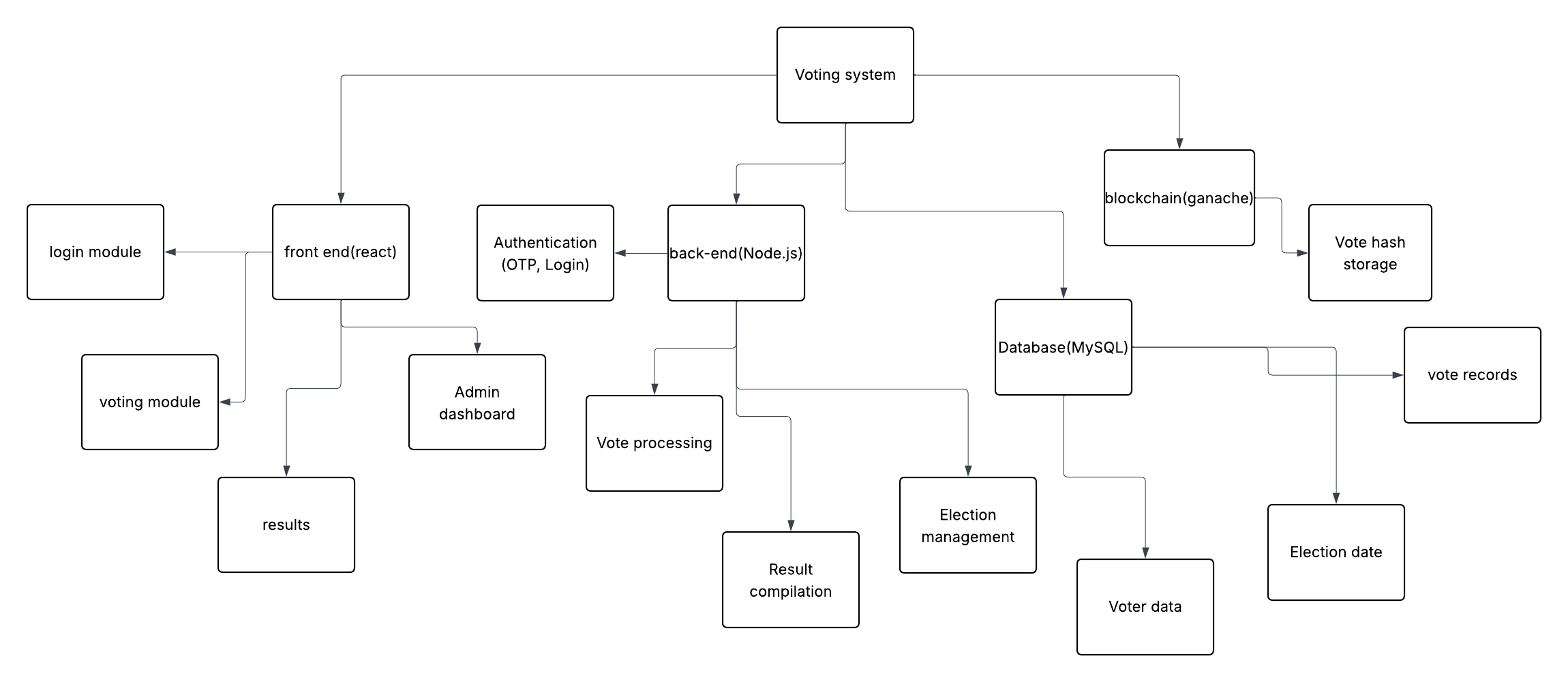
e.g. vote\_id=401, voter\_id=101, candidate\_id=201, hash="0x(jhq6hK3...", timestamp="2025-05-07 10:00:00".

The keys assist in prevention of duplicate votes which is a key objective for the system in being tamper-proof.

They also assist in linking data logically in the system.

### 5.5.6 Block Diagram/Structured/Modular Chart: System Map

The block diagram below shows how the system parts fit together.



### 5.5.7 Process / Program Design / UML

This section maps how the system works, using flowcharts to show step-by-step processes, how modules connect and how individual functions operate.

#### 5.5.7.1 System Flowchart

The system flowchart shows the entire voting process from start to finish, like a roadmap of how voters and admins use the system. It covers the user’s processes (login, vote, verify results) and backend processes (data storage, blockchain hashing).

Using the system as a user involves the following processes:

**Start**: User (voter/admin) opens the React app.

**Login**: Enter ID/password/OTP.

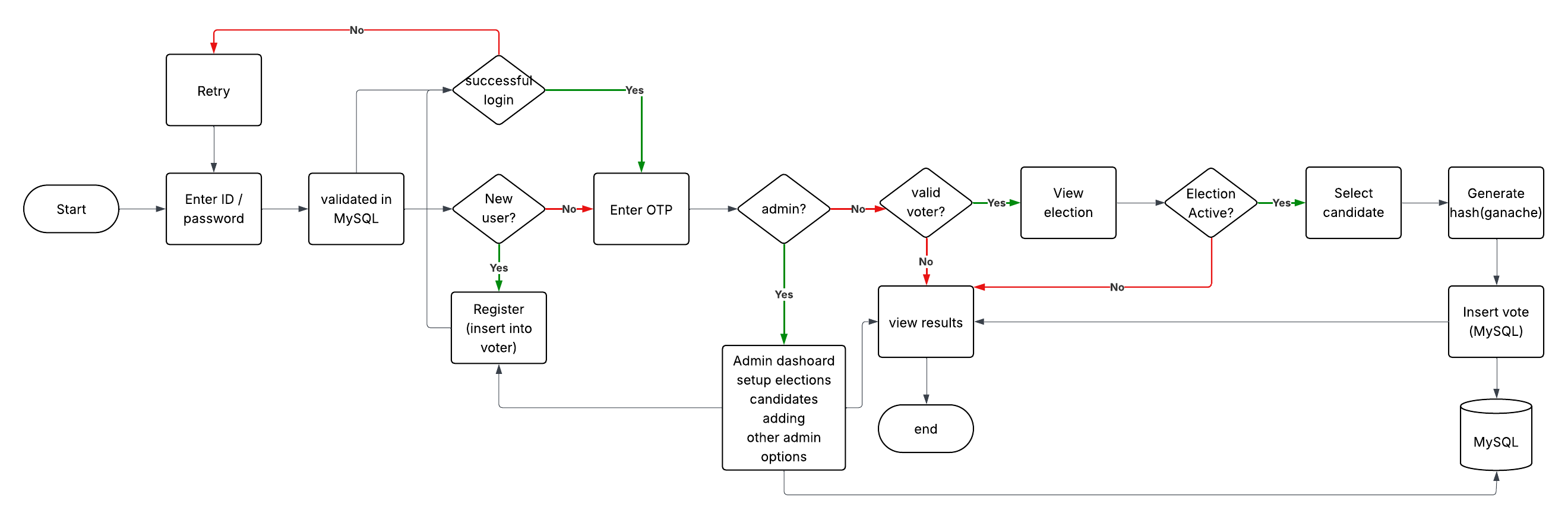
**Validate**: Node.js checks Voter table in MySQL.

**Decision**: if a new user, then Register (form inserts into Voter).

**Vote**: Voter selects candidate (React form), Node.js generates hash (Ganache), inserts into Vote (MySQL).

**Verify Results**: Voter/admin views results (React), checks hash on blockchain.

**End**: User logs out or session ends.



#### 5.5.7.2 Program Flowchart: Integrating All Modules

The program flowchart shows how the system modules are connected and how they work together. It involves integration of all components to make a system that flows cohesively.

The purpose of the program flowchart includes

* Showing how modules interact via react and node.js
* Highlighting data flow
* Ensuring that the system has smooth, secure and verifiable processes.

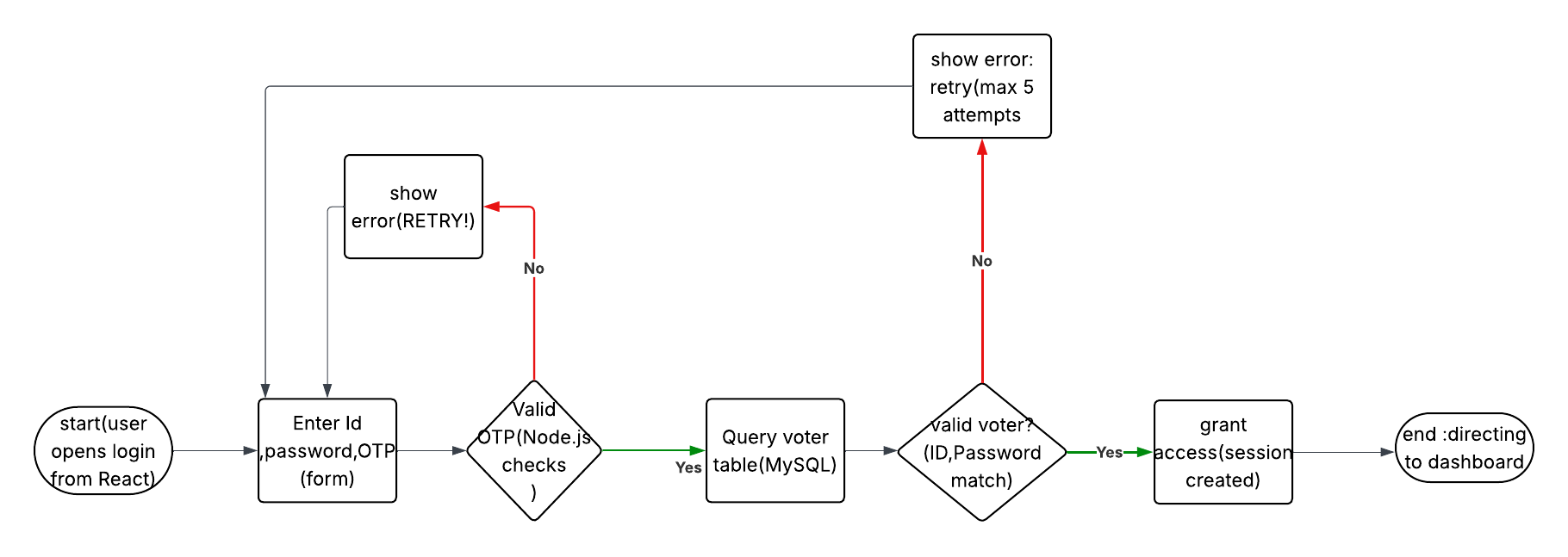
**Program flowchart**

* **Modules**
  + **Authentication:** login, registration with the use of OTP
  + **Voting:** candidate selection, hash generation.
  + **Verification:** Results display and verifying, blockchain hash check.
  + **Admin:** Election setup, candidate setup, organization setup
* **Flow**
  + **Start:** the web page is initialized (React)
  + **Authentication:** users log in or login with their details (React → Node.js → Ganache/MySQL)
  + **Decision:** option for admin or voter  
    Admin- admin dashboard  
    voter- voting module if eligible to vote.
  + **Voting module:** selection of candidate and submitting the vote (React → Node.js → Ganache/MySQL).
  + **Verification module:** viewing of results, verifying hash (React → Ganache).
  + **End:** the user logs out or session ends.

#### 5.5.7.3 Modular Program Flowcharts

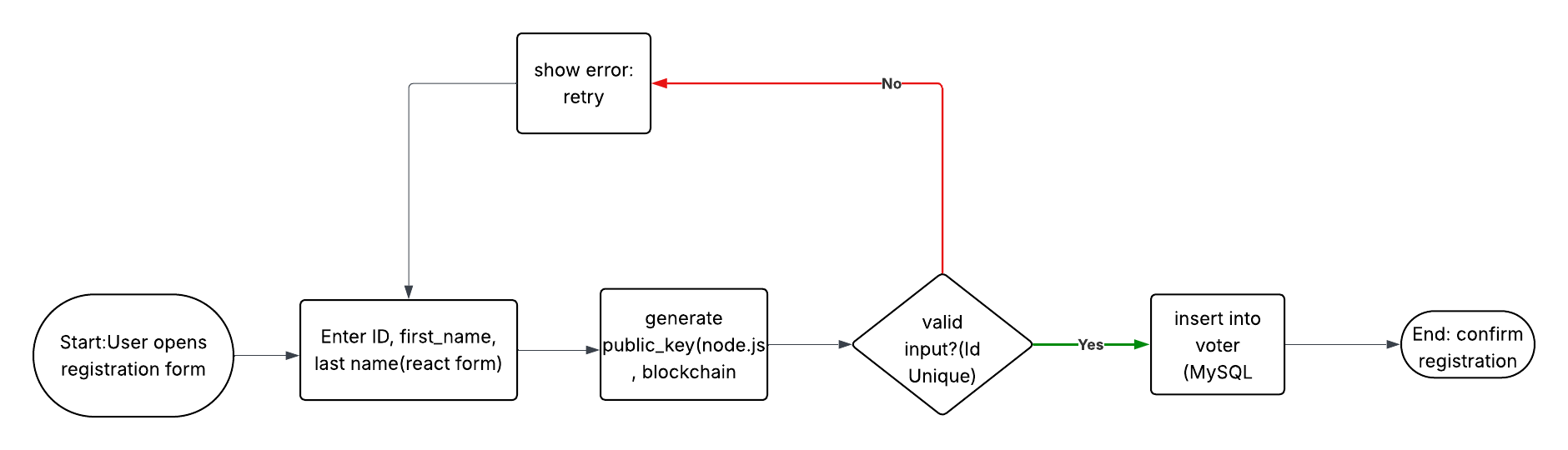
The modular flowchart breaks down critical functions involved in the system to individual flowchart. Each flowchart has a specific modules or tasks. These aids during coding in the next chapter which will ensure clarity, security and efficiency. The pseudocode for the modules is include for easier understanding of the program

Login



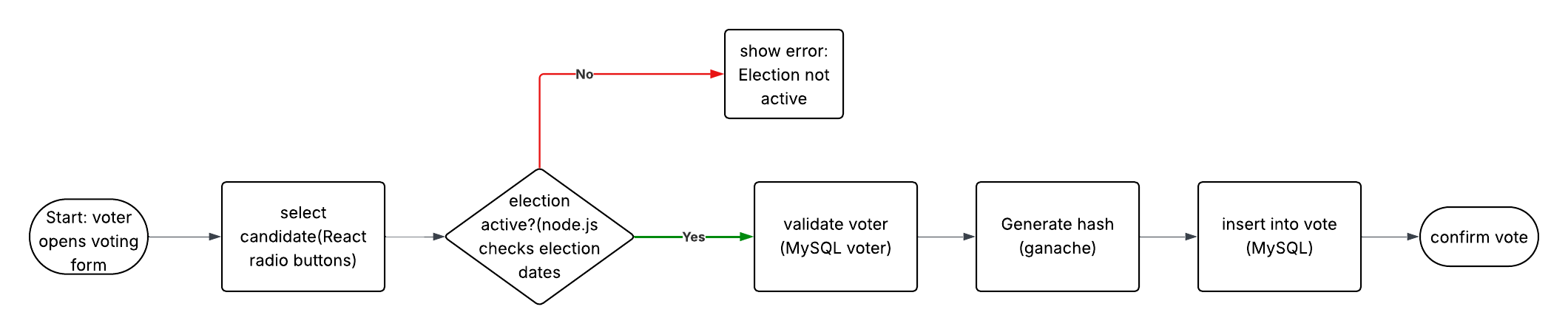
FUNCTION Login(id, password, otp)  
 IF ValidateOTP(otp)  
 IF VoterExists(id, password) IN Voter  
 RETURN "Login successful"  
 ELSE  
 RETURN "RETRY"  
 END IF  
 ELSE  
 RETURN "Invalid OTP"  
 END IF  
END FUNCTION

Voter registration



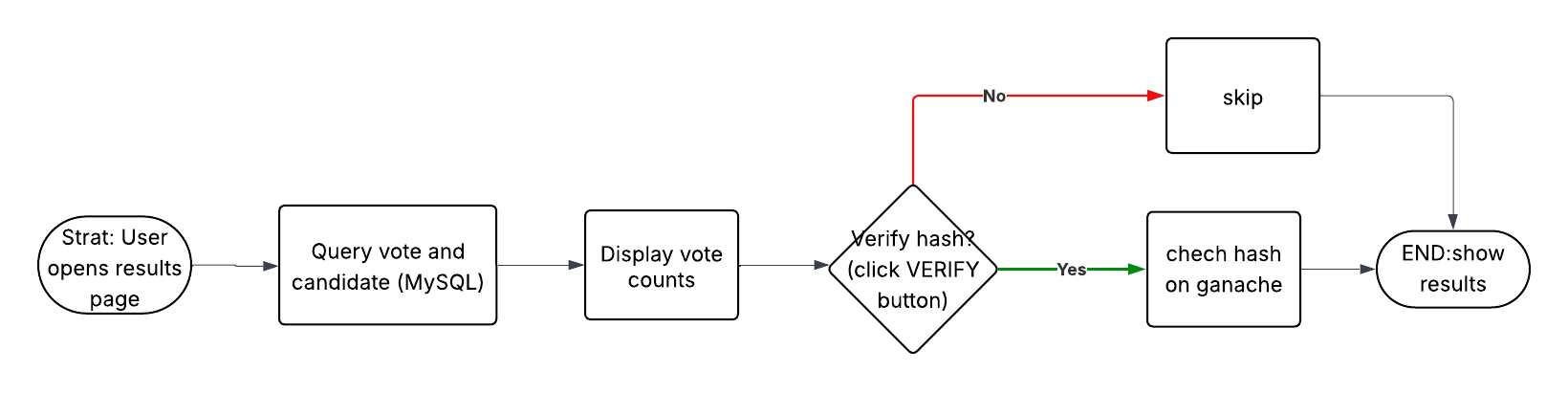
FUNCTION Register(id, first\_name, last\_name, org\_id)  
 public\_key = GeneratePublicKey()  
 IF IsUnique(id, public\_key)  
 INSERT INTO Voter (id, first\_name, last\_name, public\_key, org\_id)  
 RETURN "Registration successful"  
 ELSE  
 RETURN "ID or public key exists"  
 END IF  
END FUNCTION

Vote casting



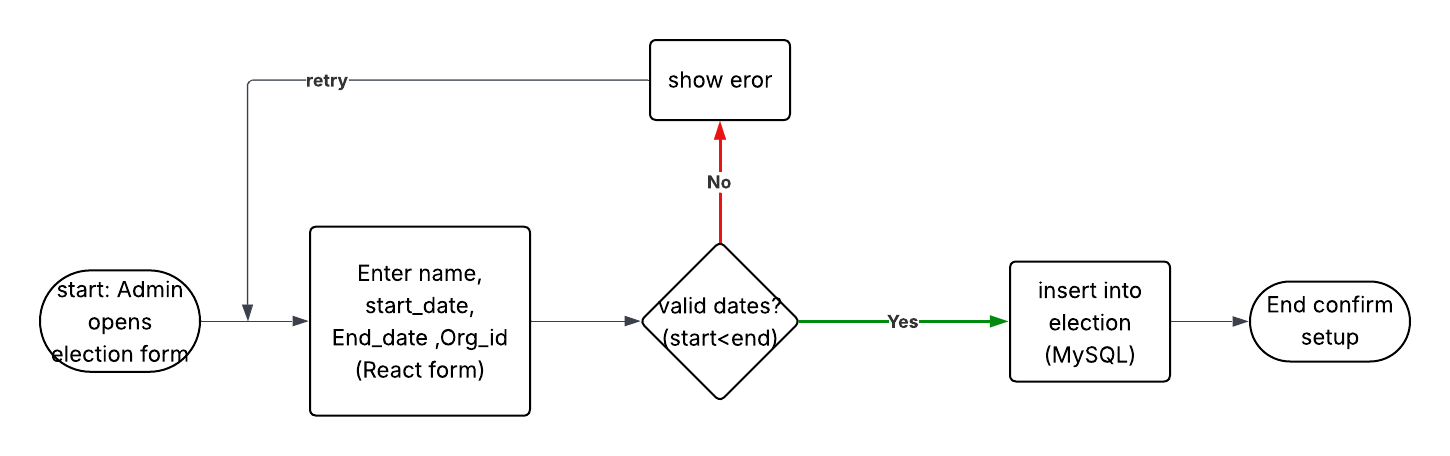
FUNCTION CastVote(voter\_id, candidate\_id)  
 IF IsVoterValid(voter\_id) AND IsElectionActive()  
 hash = GenerateBlockchainHash(voter\_id, candidate\_id)  
 INSERT INTO Vote (voter\_id, candidate\_id, hash, NOW())  
 StoreHashOnGanache(hash)  
 RETURN "Vote recorded"  
 ELSE  
 RETURN "Invalid voter or election"  
 END IF  
END FUNCTION

Result verification



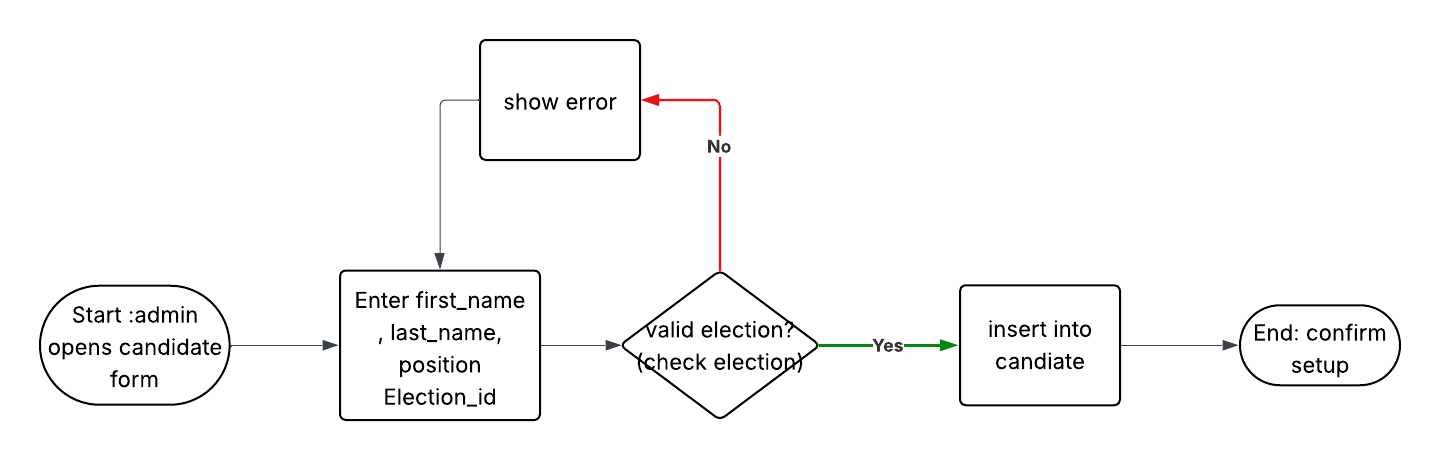
FUNCTION VerifyResults(election\_id)  
 results = SELECT c.first\_name, c.last\_name, COUNT(v.vote\_id)  
 FROM Candidate c LEFT JOIN Vote v ON c.candidate\_id = v.candidate\_id  
 WHERE c.election\_id = election\_id  
 GROUP BY c.candidate\_id  
 IF UserClicksVerify()  
 VerifyHashOnGanache(results.hash)  
 END IF  
 RETURN results  
END FUNCTION

Election setup



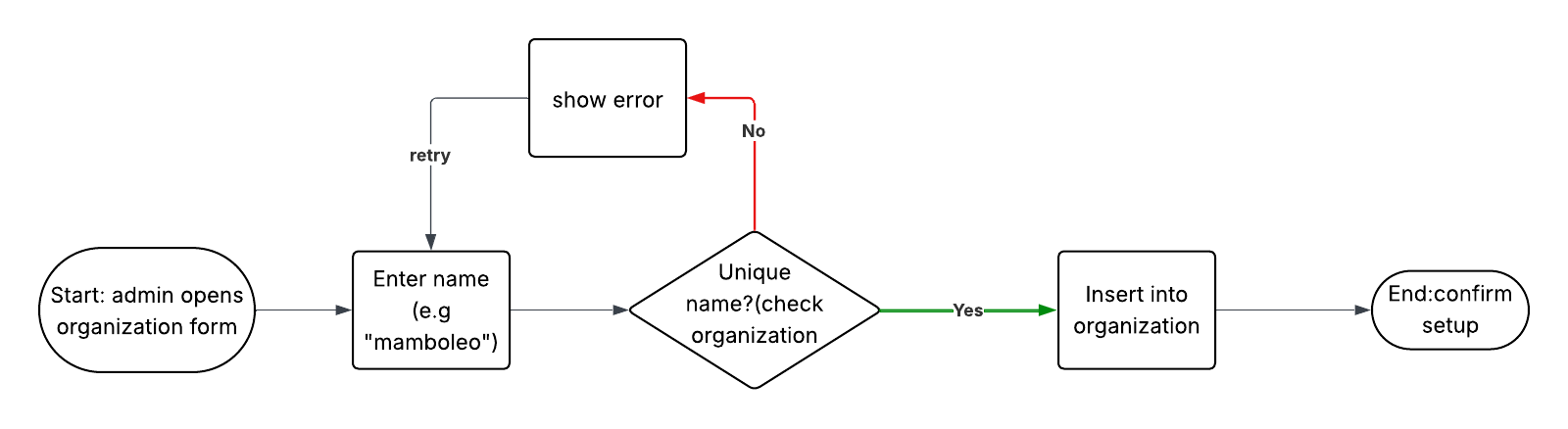
FUNCTION SetupElection(name, start\_date, end\_date, org\_id)  
 IF start\_date < end\_date AND IsValidOrg(org\_id)  
 INSERT INTO Election (name, start\_date, end\_date, org\_id)  
 INSERT INTO Log (action, user\_id, timestamp, details)  
 VALUES ('Created election', 1, NOW(), name)  
 RETURN "Election created"  
 ELSE  
 RETURN "Invalid dates or organization"  
 END IF  
END FUNCTION

Candidate setup



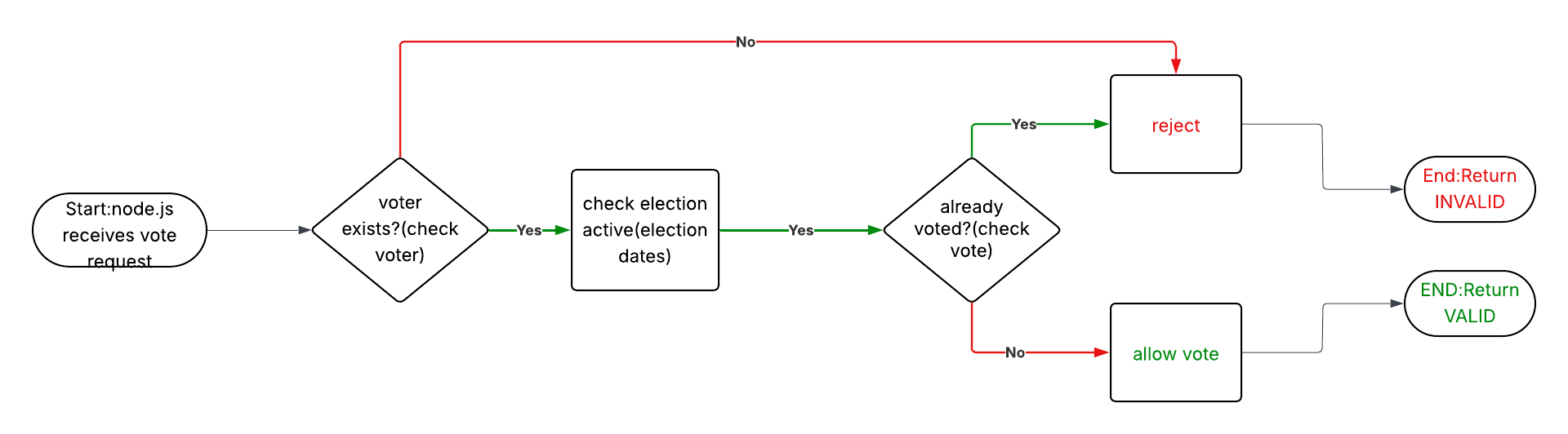
FUNCTION SetupCandidate(first\_name, last\_name, position, election\_id)  
 IF IsValidElection(election\_id)  
 INSERT INTO Candidate (first\_name, last\_name, position, election\_id)  
 INSERT INTO Log (action, user\_id, timestamp, details)  
 VALUES ('Added candidate', 1, NOW(), first\_name + ' ' + last\_name)  
 RETURN "Candidate added"  
 ELSE  
 RETURN "Invalid election"  
 END IF  
END FUNCTION

Organization setup



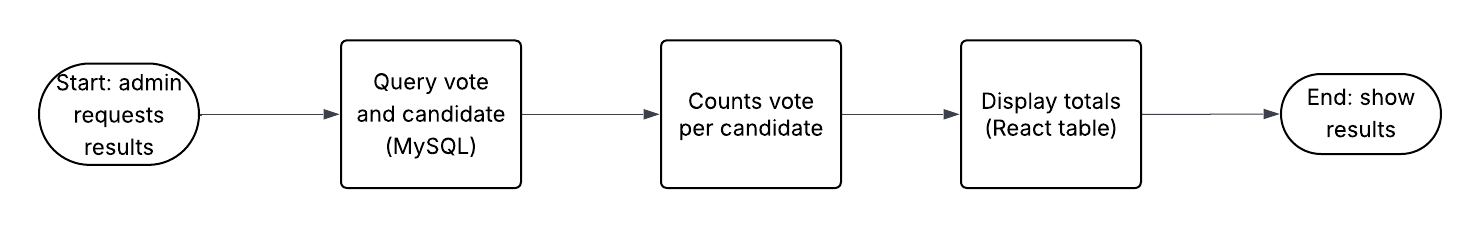
FUNCTION SetupOrganization(name)  
 IF IsUniqueName(name)  
 INSERT INTO Organization (name)  
 INSERT INTO Log (action, user\_id, timestamp, details)  
 VALUES ('Created organization', 1, NOW(), name)  
 RETURN "Organization created"  
 ELSE  
 RETURN "Name exists"  
 END IF  
END FUNCTION

Vote validation



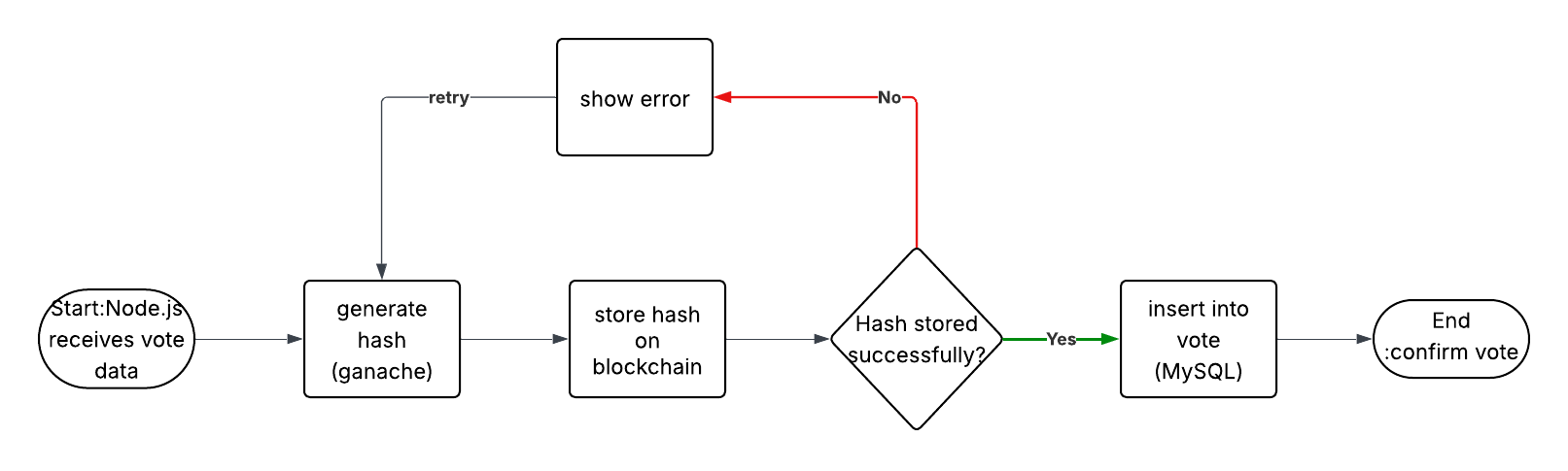
FUNCTION ValidateVote(voter\_id, candidate\_id)  
 IF VoterExists(voter\_id) AND ElectionActive(candidate\_id)  
 IF NOT HasVoted(voter\_id, election\_id)  
 RETURN "Valid"  
 ELSE  
 RETURN "Already voted"  
 END IF  
 ELSE  
 RETURN "Invalid voter or election"  
 END IF  
END FUNCTION

Result compilation



FUNCTION CompileResults(election\_id)  
 results = SELECT c.first\_name, c.last\_name, c.position, COUNT(v.vote\_id)  
 FROM Candidate c LEFT JOIN Vote v ON c.candidate\_id = v.candidate\_id  
 WHERE c.election\_id = election\_id  
 GROUP BY c.candidate\_id  
 RETURN results  
END FUNCTION

Blockchain integration



FUNCTION StoreVoteOnBlockchain(voter\_id, candidate\_id)  
 hash = GenerateHash(voter\_id, candidate\_id)  
 IF StoreHashOnGanache(hash)  
 INSERT INTO Vote (voter\_id, candidate\_id, hash, NOW())  
 INSERT INTO Log (action, user\_id, timestamp, details)  
 VALUES ('Stored vote', 1, NOW(), 'Hash: ' + hash)  
 RETURN "Vote stored"  
 ELSE  
 RETURN "Blockchain error"  
 END IF  
END FUNCTION

**References**

1. N. Kshetri and J. Voas, “Blockchain-enabled e-voting,” IEEE Software, pp. 1–5, 2018.
2. R. Tso, Z.-Y. Liu, and J.-H. Hsiao, “Distributed e-voting and e-bidding systems based on smart contract,” Electronics, vol. 8, pp. 1–22, 2019.
3. M. Marchesi, L. Marchesi, and R. Tonelli, “An agile software engineering method to design blockchain applications,” vol. 2016, pp. 1–8, 2016.
4. Ashish Balti, Abhishek Prabhu, Sanskar Shahi, Shrutika D, Vrajesh M, “A decentralized and immutable E-voting system using blockchain.” 2023
5. S. Al-Maaitah, M. Qatawneh and A. Quzmar, "E-voting system based on blockchain technology: A survey", *2021 International Conference on Information Technology (ICIT)*, pp. 200-205, 2021, July. <https://ieeexplore.ieee.org/document/9491734>
6. Lehoucq f, “electoral fraud: causes types and consequences. (2003)
7. D. J. Khoury, E. F. Kfoury, A. Kassem and H. Harb, "Decentralized Voting Platform Based on Ethereum Blockchain", *2018 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET)*, pp. 1-6, 2018.

# **Appendix**

