IT465 - BlockChain Project Report

Blockchain-Based Voting System with Zero-Knowledge Proofs

Submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

by

Rounak Jain (211IT055)

Verma Ayush (211IT079)

under the guidance of

Dr.Bhawana Rudra

DEPARTMENT OF INFORMATION TECHNOLOGY

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

SURATHKAL, MANGALORE - 575025

November, 2024

DECLARATION

We hereby *declare* that the Major Project-I Work Report entitled ***"Blockchain-Based Voting System with Zero-Knowledge Proofs"*** , which is being submitted to the **National Institute of Technology Karnataka, Surathkal**, for the award of the Degree of Bachelor of Technology in Information Technology, is a *bonafide report of the work carried out by us.* The material contained in this Course Project Report has not been submitted to any University or Institution for the award of any degree.

Name of the Student (Registration Number) with Signature

(1) Rounak Jain (211IT055)

(2) Verma Ayush (211IT079)

Department of Information Technology
Place : NITK, Surathkal
Date :07/11/2024

CERTIFICATE

This is to *certify* that the Major Project Work Report entitled ***"Blockchain-Based Voting System with Zero-Knowledge Proofs"*** submitted by

Name of the Student (Registration Number)

(1) Rounak Jain (211IT055)

(2) Verma Ayush (211IT079)

as the record of the work carried out by them, is *accepted as the B.Tech. Course Project work report submission* in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Information Technology in the Department of Information Technology, NITK Surathkal.

Signature of Course Project Guide on 07-11-2024

Dr.Bhawana Rudra

Assistant Professor

Department of Information Technology

NITK Surathkal-575025

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my course project supervisor, Dr. Bhawana Rudra, for her invaluable guidance, support, and encouragement throughout the project. Her expertise and insights were crucial in shaping the direction and execution of the project, and her feedback helped me overcome various challenges. Thank you!

ABSTRACT

This project proposes the development of an 'Blockchain-Based Voting System with Zero-Knowledge Proofs' to enhance the transparency, security, and privacy of electronic voting systems. By leveraging blockchain technology, the voting process achieves immutability and decentralized transparency, allowing votes to be securely recorded on an unalterable public ledger. Integrating zero-knowledge proofs adds an additional privacy layer, ensuring voter anonymity by allowing users to verify their votes without revealing any identifying information. This dual approach addresses traditional voting concerns, such as fraud, tampering, and voter privacy, providing a secure and transparent voting solution. The application is implemented using Ethereum's Solidity smart contracts, providing features for vote casting, candidate management, and results declaration. The proposed system is an advancement in modern electronic voting by combining the immutability of blockchain with the privacy guarantees of ZKP, thus ensuring a robust, verifiable, and privacy-preserving election process.

Keywords— Blockchain, Zero-Knowledge Proof (ZKP), Online Voting System, Smart Contracts, Ethereum, Solidity, Privacy, Security, Decentralized Voting

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Overview

The digitization of voting systems has become increasingly important in modern societies as they seek to streamline and secure electoral processes. Traditional voting methods are often prone to issues such as voter fraud, tampering, lack of transparency, and limited accessibility. To address these challenges, this project presents an online voting application that utilizes blockchain technology and zero-knowledge proof (ZKP) protocols. Blockchain, a decentralized and immutable ledger, ensures that all votes are securely recorded and publicly verifiable without central authority intervention. Zero-knowledge proofs further enhance this system by allowing voters to prove they have voted without disclosing their identity or any vote-specific details, thereby preserving privacy.

This application is implemented using Ethereum's Solidity smart contracts, which are self-executing code on the blockchain that automatically enforce the rules of the election process. This approach creates a transparent, tamper-resistant voting system where each vote is verifiable, yet voter identities remain private. By combining blockchain's transparency with ZKP's privacy assurances, this system aims to solve the core issues that plague traditional voting systems

1.2 Motivation

The motivation for developing an online voting application using blockchain and zero-knowledge proof (ZKP) stems from the need to address critical weaknesses in traditional voting systems, particularly around transparency, security, and voter privacy. Conventional voting processes often lack transparency, leaving room for doubts about the integrity and accuracy of results. Issues such as vote tampering, lack of accountability, and limited auditability diminish public trust and compromise the legitimacy of democratic processes. By leveraging blockchain's immutable and decentralized ledger, this application creates a transparent voting environment where

each vote is securely recorded and publicly verifiable, effectively mitigating risks of tampering or fraud. Additionally, voter privacy remains a paramount concern in digital voting systems, as personal data and voter choices are often vulnerable to exposure or misuse. Integrating zero-knowledge proof protocols allows this application to preserve voter anonymity by enabling users to prove they have voted without revealing their specific vote or personal information, thus safeguarding privacy and preventing voter coercion. Beyond enhancing security and privacy, an online voting system also improves accessibility, allowing voters to participate remotely and conveniently, which can boost voter turnout and inclusivity. Overall, this project combines the strengths of blockchain and ZKP to create a trustworthy, resilient voting platform, showcasing how advanced technologies can support fair, private, and transparent electoral processes.

CHAPTER 2

LITERATURE REVIEW

2.1 Background and Related Works

The paper [1]combines blockchain with Zero-Knowledge Proofs (ZKPs) to ensure voter privacy and vote transparency. Blockchain guarantees immutability and transparency, while ZKPs verify votes without revealing personal data. It proposes a tamper-proof e-voting system to prevent vote manipulation. The system ensures privacy and transparency in the election process. The approach addresses the challenges of maintaining voter anonymity in digital voting.

The paper [2] presents a two-phase e-voting system using ZKPs for verifying voter eligibility without revealing personal details. ZKPs are applied during both voter registration and vote casting to ensure privacy. It guarantees the integrity of votes without exposing any voter's information. The system prevents fraud and ensures that votes remain anonymous. The approach is scalable and complies with privacy standards.

The paper [3] proposes an e-voting system that combines blockchain and ZKPs for end-to-end verifiability. Blockchain ensures decentralized data storage while ZKPs verify votes without disclosing voter choices. It provides a secure and transparent method for vote casting and tallying. The system eliminates single points of failure, improving election reliability. It aims to improve voter trust in the integrity of digital elections.

The paper [4] introduces a blockchain-based carpooling platform that emphasizes user privacy and security. Blockchain technology ensures a transparent, tamper-proof ledger for carpool transactions without centralized control. To protect user privacy, Zero-Knowledge Proofs (ZKPs) verify user credentials (e.g., ownership or eligibility) without exposing sensitive details. This system reduces the dependency on intermediaries, offering a more secure, trustless, and efficient carpooling experience. The study highlights ZKPs and blockchain's potential for secure, decentralized applications in the sharing economy.

The paper [5] Ethereum smart contracts, combined with ZKPs, to create a decentralized, self-tallying voting system. The system automates vote tallying, ensuring transparency and tamper resistance. ZKPs verify vote validity while keeping voter identities anonymous. Blockchain technology prevents vote manipulation and fraud. The solution addresses efficiency and transparency challenges in online elections.

The paper [6] integrates homomorphic encryption with ZKPs for secure vote casting and tallying. Votes are encrypted during both casting and counting, ensuring privacy. Homomorphic encryption allows votes to be tallied without decryption. ZKPs validate the authenticity of encrypted votes without revealing their content. The system is implemented using smart contracts for automation and security.

The paper [7] uses Zero-Knowledge Rollups and ZKPs to enhance scalability and privacy in blockchain-based voting systems. The protocol ensures that votes are securely verified while maintaining voter anonymity. Blockchain provides a decentralized platform, while ZKPs ensure transparent and tamper-proof results. The approach supports large-scale elections without compromising security. The system aims to solve scalability issues in digital voting.

The paper [8] investigates the combination of blockchain and ZKPs to create secure, transparent, and tamper-resistant online voting systems. Blockchain's decentralization ensures data integrity, while ZKPs validate votes without exposing personal details. It highlights the ability of this technology to prevent vote manipulation. The system offers both transparency and voter anonymity. This solution addresses major concerns in traditional voting systems.

2.2 Outocome of Literature Review

The literature on secure and decentralized applications leveraging blockchain and Zero-Knowledge Proofs (ZKPs) reveals significant advancements in privacy and security. Several studies focus on using blockchain to decentralize data storage and improve transaction transparency. For instance, Yu et al. (2021) and Wang et al. (2020) showcase the potential of ZKPs in maintaining user privacy by validating transactions without disclosing sensitive information. These systems enhance trust and security in contexts like e-voting, where transparency and data immutability are

critical. Similarly, Almasri et al. (2022) emphasize using ZKPs to establish verifiable systems that maintain data integrity without central intermediaries.

Other studies, such as Choudhury et al. (2020) and Rani et al. (2021), explore the integration of blockchain and cryptography in creating resilient, privacy-preserving applications. Their systems showcase how ZKPs help protect user credentials, thereby reducing reliance on intermediaries and enhancing trust. Goel et al. (2022) extend these principles to a carpooling application, demonstrating blockchain's adaptability across domains by ensuring secure, trustless peer-to-peer interactions. Together, these studies underscore the adaptability of ZKPs and blockchain in creating decentralized solutions for a wide array of applications, particularly in environments where user privacy and transaction integrity are paramount.

2.3 Problem Statement

his project aims to develop a secure, anonymous, and tamper-proof online voting system using blockchain and Zero-Knowledge Proofs (ZKP).

2.4 Objectives of the Project

(1) Implement a secure online voting system using blockchain to ensure transparency and immutability of votes.

(2) Integrate Zero-Knowledge Proofs (ZKP) to verify voter eligibility and vote validity while maintaining voter anonymity.

(3) Prevent vote tampering and double voting through decentralized and tamper-proof blockchain records.

CHAPTER 3

PROPOSED METHODOLOGY

3.1 System Architecture

The system architecture of the voting contract as shown in Figure 3.2.1 is designed to ensure secure, private, and efficient elections using blockchain and Zero-Knowledge Proofs (ZKPs). The Voter generates a ZKP to maintain privacy and submits it along with the chosen candidate's ID to the Voting Contract. The Voting Contract then forwards the proof to the Verifier Contract for validation. If the proof is valid, the Voting Contract updates the vote count for the candidate and records the voter's participation. The Owner (Admin) as shown in Figure 3.2.2 manages the election by adding candidates before the election begins and ending the election to finalize the results. This architecture ensures election integrity, privacy, and transparency, leveraging blockchain's immutability and ZKPs' privacy-preserving features to guarantee a trustless, tamper-proof voting process.

3.2 State Variables

The contract maintains essential state variables:

owner: the contract creator, who has the exclusive right to start and end the election.

electionEnded: a boolean flag indicating whether the election is active or has ended.

verifier: an instance of IVerifier, pointing to the deployed ZKP verifier contract.

Additionally, there are two primary data structures:

Candidate: a struct representing each candidate with attributes id, name, and voteCount.

Voter: a struct that tracks whether a voter has voted and, if so, the ID of the candidate they voted for.

Two mappings store candidate and voter information:

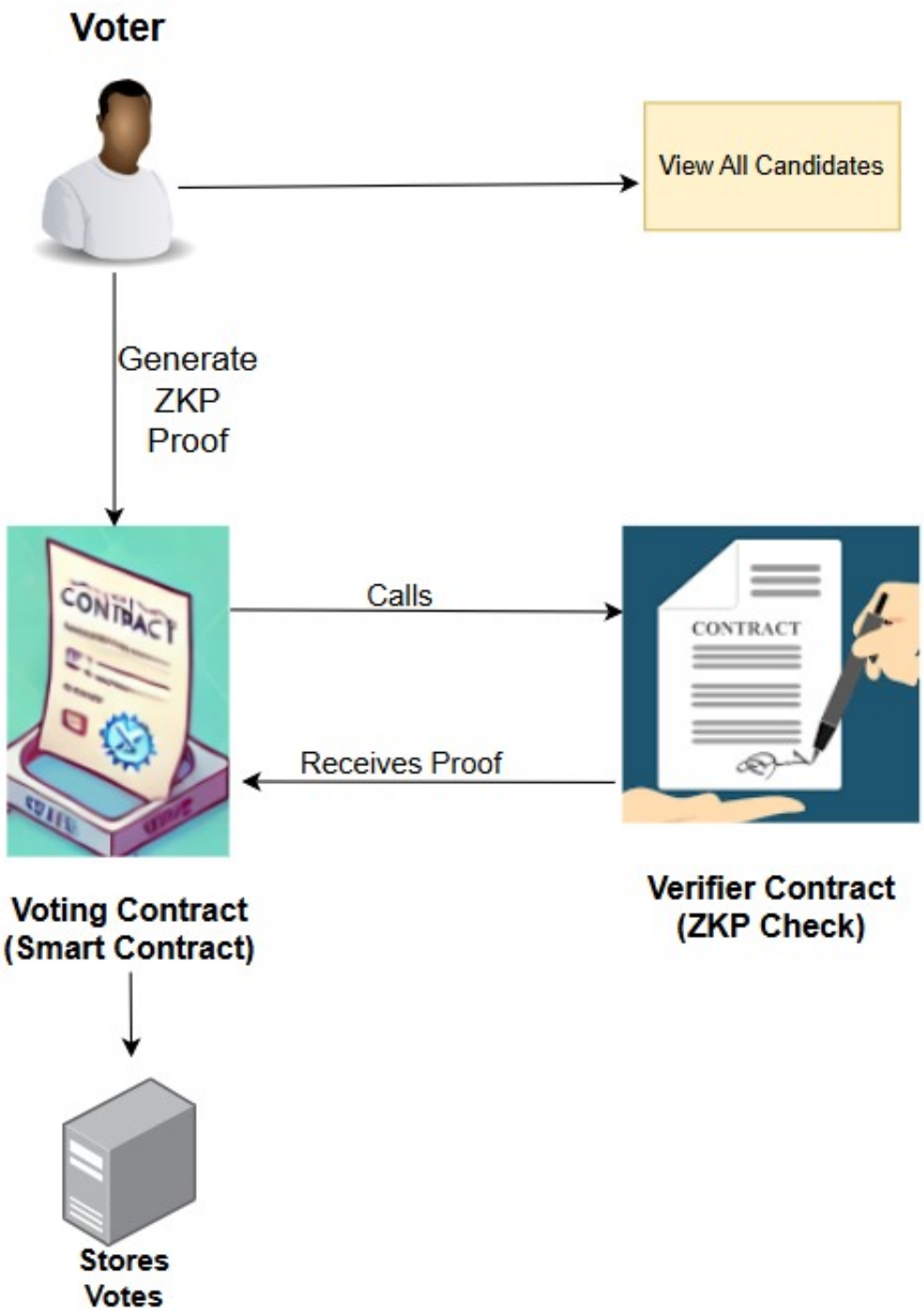

Figure 3.1.1: Voter Architecture

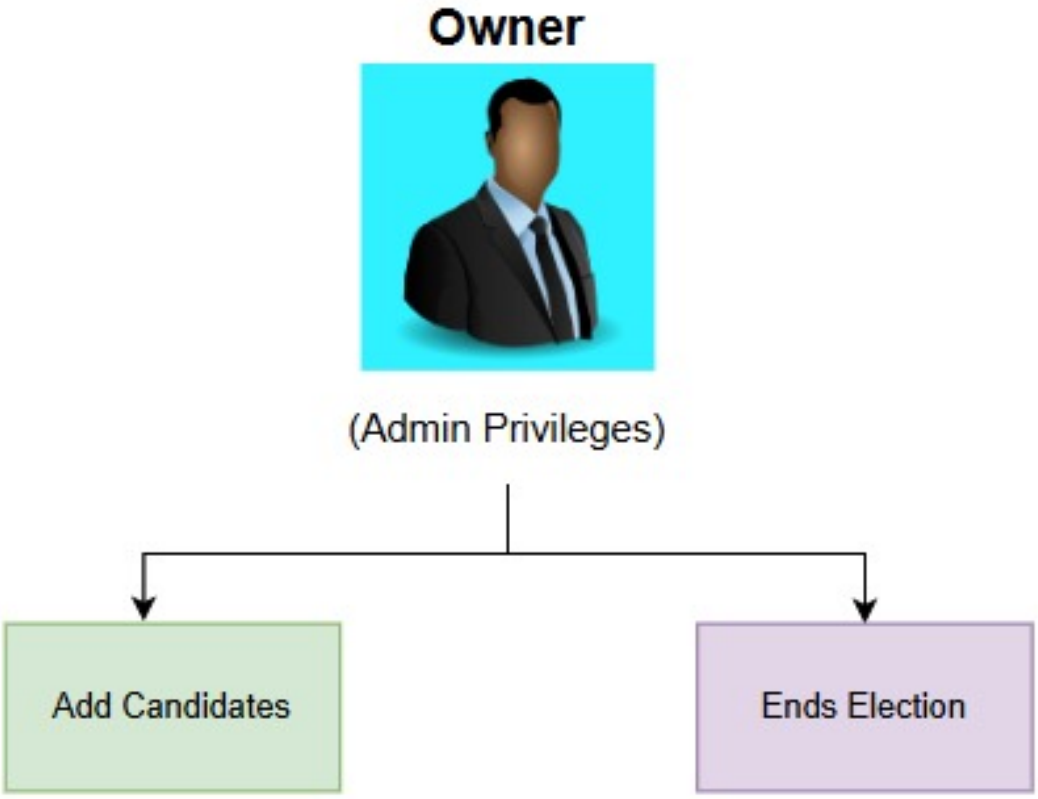

Figure 3.1.2: Owner Architecture

candidates: maps candidate IDs to Candidate structs.

voters: maps voter addresses to Voter structs, allowing each address to cast a single vote.

3.3 Modifiers for Access Control

To enforce permissions and ensure the election process is managed securely, the contract defines two modifiers:

onlyOwner: Restricts function calls to the owner of the contract, ensuring that only the contract creator can perform specific actions (e.g., adding candidates, ending the election).

electionActive: Ensures that certain functions can only be called while the election is ongoing. This modifier helps maintain the logical flow of the election, preventing actions like adding candidates or voting after the election has concluded.

3.4 Constructor

The constructor initializes key state variables:

(1)owner is set to the address of the account that deploys the contract.

(2)electionEnded is set to false, signifying that the election is active upon deployment.

(3)verifier is initialized to point to an external ZKP verifier contract address provided during deployment. after the election has concluded.

3.5 Adding Candidates

The addCandidate function, callable only by the owner during an active election, allows the owner to register candidates by name. Each candidate is assigned a unique ID, incrementing candidatesCount to track the total candidates. This function ensures that the list of candidates is finalized before voting begins, providing transparency and clarity for voters.

3.6 Retrieving All Candidates

The getAllCandidates function returns two arrays: one containing candidate names and the other containing candidate IDs. This function facilitates easy retrieval of candidate data for users, ensuring they can review the candidates and their IDs before voting. By allowing voters to access candidate information without interacting with mappings directly, this approach enhances user experience.

3.7 Casting a Vote with ZKP Verification

The vote function in the contract enables a user to cast a vote for a specific candidate, with each vote validated through a zero-knowledge proof (ZKP) to ensure integrity and privacy. To prevent double voting, the function first checks if the voter has already participated by inspecting the hasVoted flag in the voter's record. Next, it validates the candidate's ID to ensure it corresponds to an eligible candidate. Once these conditions are met, the function proceeds with ZKP verification by calling verifier.verifyProof with the proof parameters supplied by the user. If the verifyProof call returns true, the proof is deemed valid, and the contract registers the vote; otherwise, it reverts with an error message, rejecting any invalid submissions. Upon successful proof verification, the contract updates the voter's status in the voters mapping, recording their choice, and increments the chosen candidate's voteCount. This structured approach ensures that only valid, unique votes are counted, preserving the election's fairness and security.

3.8 Fetching Total Votes for a Candidate

The totalVotes function provides the total votes for a specified candidate ID. This read-only function allows any user to check the vote count of a candidate during or after the election. This transparency promotes trust, enabling participants to verify that votes are accurately counted.

3.9 Ending the Election

The endElection function, restricted to the owner, allows the contract creator to conclude the election. By setting electionEnded to true, this function prevents further voting, ensuring the election results are finalized. An ElectionEnded event is emitted to signal the election's conclusion, allowing off-chain systems or user interfaces to react to this change.

3.10 Determining the Winner

The getWinner function returns the name and vote count of the candidate with the highest votes. It first checks that the election has ended to prevent prematurely revealing results. It then iterates through all candidates to identify the one with the highest voteCount. This design avoids real-time race conditions during voting by only assessing the winner after the election is closed, preserving the integrity of the voting process.

CHAPTER 4

RESULTS AND ANALYSIS

4.1 Implentation Outputs

The deployment and interaction with the voting contract were successfully conducted using MetaMask and Remix as shown in Figure 4.1.1. MetaMask was utilized to create both the owner (admin) and multiple voter accounts, enabling seamless switching for testing purposes.

The contract was deployed via the Remix IDE by connecting to MetaMask's injected Web3 environment as shown in Figure 4.1.2, allowing the contract owner to add candidates and manage the election lifecycle as shown in Figure 4.1.3. Voters cast their votes by submitting candidate IDs along with Zero-Knowledge Proofs (ZKPs), which were validated by the contract to ensure secure, anonymous voting. Once the election concluded, the contract's getWinner function accurately identified and returned the winning candidate based on the recorded votes as shown in Figure 4.1.4. This process verified that the voting contract performs its intended functions effectively, ensuring a transparent and secure election.

4.2 Analysis

The effectiveness of the voting contract system was evaluated across several critical aspects, as outlined below:

(1)ZKP Validation: The Zero-Knowledge Proof (ZKP) mechanism integrated within the contract was successful in validating votes while ensuring voter privacy. Only votes accompanied by valid proofs were counted, effectively maintaining the integrity of the voting process without revealing the voter's identity or choice. This privacy-preserving feature of ZKPs was central to the security of the election system.

(2)Vote Counting: The contract accurately maintained and updated the vote count for each candidate. The totalVotes function allowed for real-time tracking of the votes, providing precise and transparent results at any point during the election.

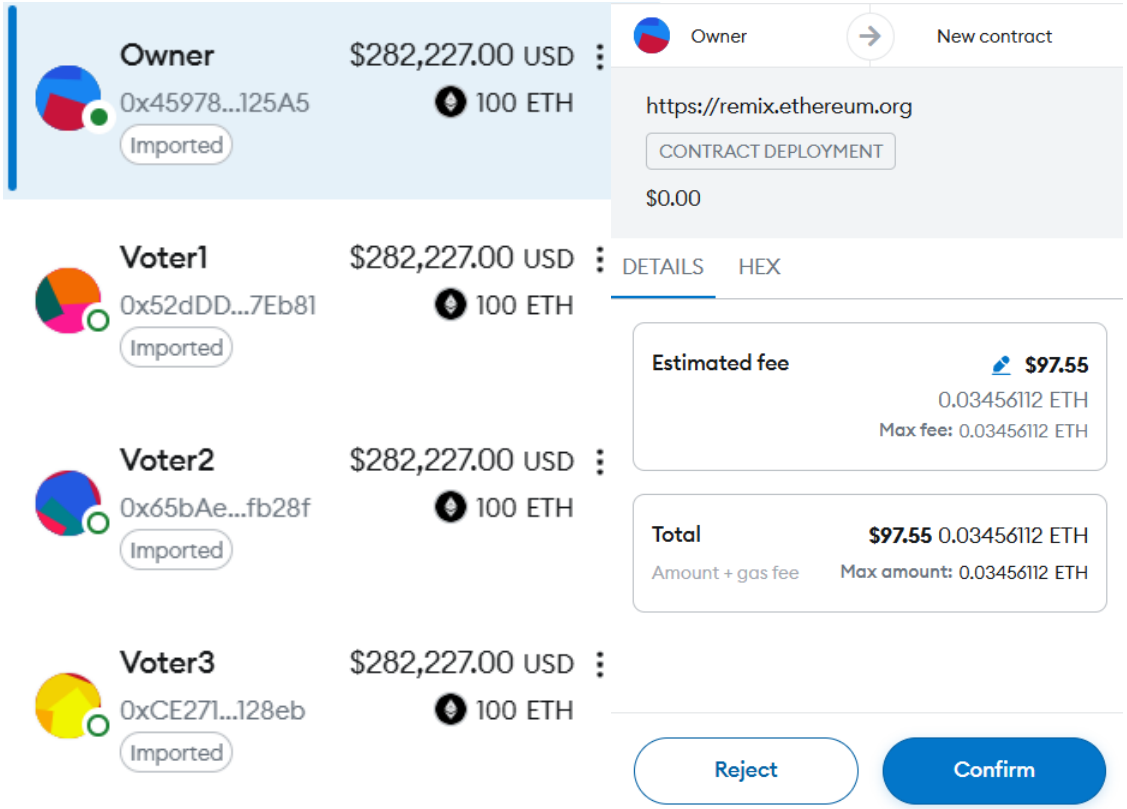

Figure 4.1.1: Owner and Voter Creation

Figure 4.1.2: Metamask Deployment

This ensures that the vote count could be independently verified by any participant or observer.

(3)Election Integrity: By utilizing blockchain technology, the election process was secured against tampering and unauthorized modifications. Once a vote was cast, it became immutable and could not be altered, ensuring the integrity and trustworthiness of the election results. The transparency of the blockchain also allowed all participants to verify the election's legitimacy.

(4)User Interaction: The contract provided an intuitive and user-friendly interface for both the election owner and voters. The election owner could register candidates and conclude the election, while voters could cast their votes efficiently. This straightforward interaction ensured that the process remained accessible and transparent for all involved parties.

(5)Privacy Preservation: The integration of ZKPs successfully preserved voter anonymity throughout the election process. Voters' identities and the candidates

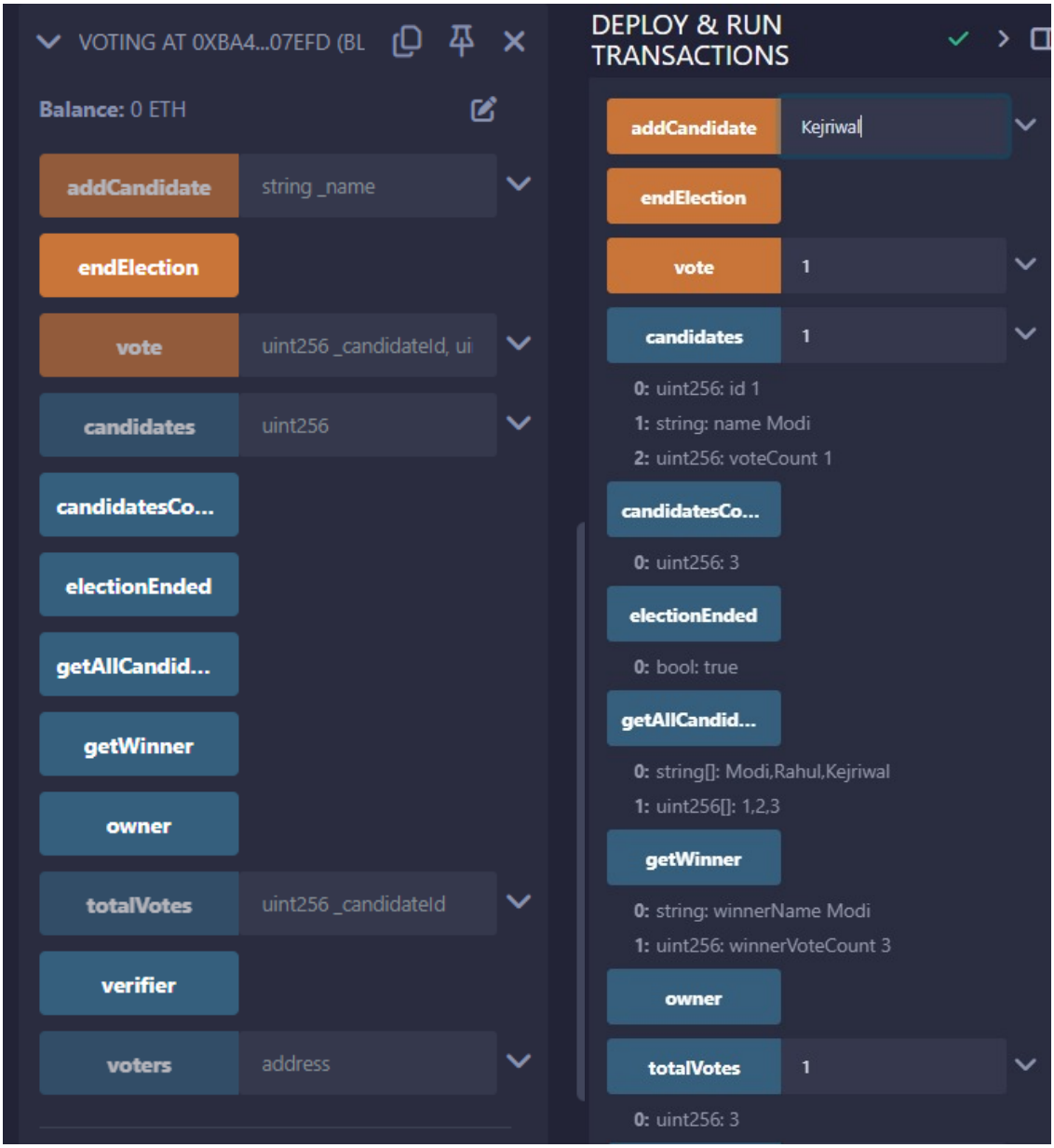

Figure 4.1.3: Deploying Smart Contract

Figure 4.1.4: Voting System Result

they voted for were kept confidential, preventing unauthorized entities from obtaining sensitive information. This added layer of privacy ensured that the election adhered to the principles of confidentiality and fairness.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

This project successfully combines blockchain and zero-knowledge proof (ZKP) technologies to create a secure, transparent, and privacy-preserving online voting system. Blockchain provides an immutable, decentralized record of votes, preventing tampering and ensuring transparency, while ZKP safeguards voter anonymity by allowing participation verification without revealing voter identities or choices. Together, these technologies address key challenges in traditional voting, offering a reliable and accessible platform for digital elections that fosters trust in the voting process.

Future work can enhance scalability and accessibility by optimizing ZKP computations or exploring alternatives like zk-STARKs, making the system more suitable for large-scale use. Additional features like secure voter authentication, user-friendly cross-platform interfaces, and regulatory compliance testing would also support real-world adoption. These advancements could extend the use of this technology beyond voting to other decentralized governance applications, demonstrating the broader potential of blockchain and ZKP in secure, privacy-focused decision-making.

REFERENCES

[1] H. Yu, J. Zhang, and Z. Li. Privacy-preserving blockchain-based e-voting system with zero-knowledge proofs. In *Proceedings of the IEEE International Conference on Blockchain and Cryptocurrency*, pages 215–221, Kyoto, Japan, April 2021. IEEE.

[2] L. Wang, Y. Wang, and J. Li. A secure voting system based on zero-knowledge proofs and blockchain. In *Proceedings of the 39th IEEE International Conference on Distributed Computing Systems*, pages 1578–1584, Singapore, June 2020. IEEE.

[3] M. Almasri, R. Patel, and F. Liu. Verifiable e-voting system using blockchain and zero-knowledge proofs. In *Proceedings of the 2022 International Conference on Artificial Intelligence and Blockchain*, pages 233–238. ACM, November 2022.

[4] Saksham Goel, Sarvesh V. Sawant, and Bhawana Rudra. Secure decentralized carpooling application using blockchain and zero-knowledge proof. In *Proceedings of the National Institute of Technology Karnataka*, Department of Information Technology, National Institute of Technology Karnataka, India, 2022.

[5] P. Rani, S. Gupta, and A. Jain. Decentralized e-voting with blockchain and zero-knowledge proofs. In *Proceedings of the International Conference on Security and Privacy in Computing and Communications*, pages 120–126, Sydney, Australia, December 2021. Springer.

[6] Q. Zhou, P. Li, and H. Zhang. Integrated blockchain and zero-knowledge proofs for secure and transparent e-voting. *Journal of Cryptographic Engineering*, 12:45–56, March 2023.

[7] S. Choudhury, A. Singh, and R. Kumar. A privacy-preserving online voting system with blockchain and zero-knowledge proofs. In *Proceedings of the IEEE Global Communications Conference*, pages 1270–1275, Taipei, Taiwan, December 2020. IEEE.

[8] R. Singh, N. Gupta, and K. Mehta. A secure online voting system using blockchain and zero-knowledge proofs for election transparency. In *Proceedings of the 8th International Conference on Emerging Technologies in Computing*, pages 105–111, Berlin, Germany, May 2022. Springer.