Transparent and Secure Electronic Voting System for Sri Lanka

A Thesis Submitted for the Degree of Master of Computer Science

N.P.L.R Pathirana

University of Colombo School of Computing

2024

DECLARATION

Name of the student: N P L R Pathirana
Registration number: 2018/MCS/061
Name of the Degree Programme: Master of Computer Science
Project/Thesis title: Secure and Transparent Electronic Voting System for Sri Lanka's Democratic Elections

1. The project/thesis is my original work and has not been submitted previously for a degree at this or any other University/Institute. To the best of my knowledge, it does not contain any material published or written by another person, except as acknowledged in the text.

2. I understand what plagiarism is, the various types of plagiarism, how to avoid it, what my resources are, who can help me if I am unsure about a research or plagiarism issue, as well as what the consequences are at University of Colombo School of Computing (UCSC) for plagiarism.

3. I understand that ignorance is not an excuse for plagiarism and that I am responsible for clarifying, asking questions and utilizing all available resources in order to educate myself and prevent myself from plagiarizing.

4. I am also aware of the dangers of using online plagiarism checkers and sites that offer essays for sale. I understand that if I use these resources, I am solely responsible for the consequences of my actions.

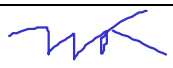
5. I assure that any work I submit with my name on it will reflect my own ideas and effort. I will properly cite all material that is not my own.

6. I understand that there is no acceptable excuse for committing plagiarism and that doing so is a violation of the Student Code of Conduct.

Signature of the Student	**Date (DD/MM/YYYY)**
	10/25/2024

Certified by Supervisor(s)

This is to certify that this project/thesis is based on the work of the above-mentioned student under my/our supervision. The thesis has been prepared according to the format stipulated and is of an acceptable standard.

	Supervisor 1	**Supervisor 2**	**Supervisor 3**
Name	Malik Silva		
Signature			
Date	26th Oct 2024		

I would like to dedicate this thesis to

my wife and my daughter,

my parents and my brother,

who motivated and encouraged me,

&

academic and non-academic staff

of

University of Colombo School of Computing,

who educated me and enabled me

to reach at this level.

ACKNOWLEDGEMENTS

I would like extend my sincere gratitude and appreciation to my supervisor Mr. KPMK Silva, senior Lecturer of the University of Colombo School of Computing for accepting to supervise, the guidance and freedom given to conduct this project. I'm also grateful to all the academic members of the University of Colombo School of Computing for their dedication and effort throughout the master's program. Lastly, I would like to extend my gratitude to my family and friends for the encouragement given to carry out this research.

ABSTRACT

Elections are the fundamental component of democracy. Elections provide a way for the people to choose their leaders or the representatives who can make the decisions on behalf of them to make the country a better place. Electronic voting is identified as a tool for making the electoral process more efficient, transparent and secure. Transparency and immutability features play the key roles in any system to gain the people's trust about the system. Blockchain distributed ledger is recognized as a technology which can adapt the above features in a secure way. This study focuses on enabling transparency through post-voting verification using the Corda, permissioned distributed ledger framework. The system uses cryptographic techniques and secure protocols to guarantee the confidentiality and integrity of votes throughout the entire electoral cycle. Each vote is encrypted and securely stored inside the vault of the corda node, and the system generates a unique reference receipt for the voter. This receipt and the password which is used in the submission of the voting time serves as the mandatory elements in the post-voting verification process. This post-voting verification not only enhances public trust for the system but also validates the integrity of the electoral process. The proposed system will ensure the fair and transparent electoral process for better democracy.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Fair elections and electoral process is important for any democratic society. The rapid advancement of the technologies has enabled the potential to have a fair and transparent electoral process by replacing the manual work with the automation. This chapter provides the overview of the project aims to investigates on how to provide a secure way to enable transparency in voting systems.

1.1 Motivation

The motivation behind this research project is the need to change the way elections are conducted in Sri Lanka. The existing manual voting system has been used by Sri Lanka a long time and it has been subject to criticism for its inherent vulnerabilities, inefficiencies and lack of transparency ("Ministry of Defence - Sri Lanka," n.d.). In the recent past, the elections were delayed and postponed due to the resource challenges faced by the country. An electronic voting system has the potential to be more secure, accurate, transparent and efficiently handle the election processes and it will reduce the time required for counting votes. Enabling verifiability through transparency is a key to building trust about the electoral process. This research project investigates on how to enable transparency in voting process through the distributed ledger in a secure and scalable way, provide a detailed design of a secure electronic voting system, and develop a prototype which could be the first step of migrating to the electronic voting solution.

1.2 Statement of the problem

Enabling transparency through post-vote verifiability.

Transparency plays a major role when it comes to building a trust in any system regardless of the industry. It has proven that the manual processes and centralized systems leads to critical issues such as vote rigging, counting inaccuracies, and delays in announcing election results. It raises concerns about its integrity and fairness of the electoral process. Some countries have already started exploring the adaptation of different types of electronic voting systems which includes mechanical and internet-based systems. But many available voting systems and proposed solutions does not have a possibility secure of post-vote verifiability. This study investigates on how to achieve transparency through post-vote verification in a decentralized system.

1.3 Research Aims and Objectives

1.3.1 Aim

Design and develop a transparent, verifiable, scalable and secure electronic voting system for Sri Lanka, mitigating the shortcomings of the existing manual process.

1.3.2 Objectives

The objectives of this study are,

- Conduct a comprehensive analysis of existing electronic voting systems and technologies.
- Gain a deep understanding of the Sri Lankan electoral process and its issues.
- Design a secure and scalable architecture of the electronic voting system.
- Integrate blockchain distributed ledger technology for transparency and auditability.
- Implement real-time vote counting and efficiency.
- Implement and integrate cryptographic protocols to ensure security and confidentiality.
- Design user-friendly interfaces and optimize the user interfaces for people with different levels of technical proficiency.

1.4 Scope

- This study focuses on the technical aspects of designing and implementing an electronic voting system.
- The proposed prototype will be a software solution.
- This system will be designed for a hybrid environment where voting will be done in a controlled environment and verification can be done in an uncontrolled environment.
- The proposed system will mainly cover the voting, counting and auditing processes.
- This study will focus on areas of privacy, transparency and security of the system.
- This study will not investigate legal considerations which might affect adopting an electronic voting system in real scenarios.

1.5 Structure of the Thesis

This section provides the overview of the study.

- Chapter 2 – Literature Review

  This section will provide the review of existing electronic voting systems and the studies conducted on such systems and related technologies.
- Chapter 3 – Methodology

  This section provides the details of the research method used and related technical details.
- Chapter 4 – Design and Implementation

  This section provides the details of the proposing system design and implementation details.
- Chapter 5 – Evaluation and Results

  This section provides the details of the evaluation of the proposed system.

CHAPTER 2

LITERATURE REVIEW

Electronic voting systems have gained significant attention in recent years globally. Some countries are using e voting systems up to different extents in their municipalities to national level elections and the system mechanisms also vary like electronic voting machines in polling stations, scanning mechanisms for counting paper ballots or internet-based voting systems. International Institute for Democracy and Electoral Assistance is an intergovernmental organization founded in 1995 which actively supports democratic institutions to develop sustainable, effective and legitimate democracies around the world ("ICTs in Elections Database | International IDEA," n.d.). According to the IDEA elections database, only 24.47% of countries use electronic voting systems and only 7.45% (14 countries) use internet-based voting systems.

2.1 Analysis of existing electronic voting systems

The existing electronic voting systems can be divided into two types of environments where the voting systems operate as controlled and uncontrolled manner.

Systems operate in controlled environments will have dedicated voting centers or devices provided by election authorities. It will have a restricted access and security measures for voter authentication and identification procedures. Majority of the countries which are using e-voting, operates the system in controlled environments. Brazil is recognized the first country to use electronic voting completely for their elections since 2000 ("Electronic Voting - Case Study: Brazil," n.d.). Those are standalone systems and not connected to the internet but it offers paper-based receipts for verifications. India also uses e voting systems and those systems audited before the elections and does not have an audit mechanism after voting done for voters which rises concerns of the verifiability. In United States there are many types of machines including touchscreens for marking the choices and optical scanners for scanning the ballots. Voter registrations are handled separately ("Electronic voting in the United States," 2024).

Systems operate in uncontrolled environments will have low level of control over voters, devices, and network access. Voters use their personal computers or smartphones to access voter portals over the internet. These systems are more rely on cryptographic security

protocols and end-to-end encryption which focuses on voter authentication an privacy assurance. Few countries including Estonia, Switzerland, Australia uses internet-based voting systems in different scales for their elections.

Estonia has used internet voting system called i-voting since 2005. It uses an e-id which includes a digital signature based on public key infrastructure to identify the residents (Pihlak, 2019; “Underlying principles of the i-voting system in Estonia - Smartmatic,” n.d.). Vote information is wrapped using the digitally signed envelop using the e-id of the voter. System enables voters to cast and change their vote multiple times prior to the election day via online or they can cast the vote at a polling station on the election day which invalidate the prior votes which demotes the vote buying and selling. System also supports validating the vote they casted which intends to ensure votes transmitted to the system is the same as the submitted vote. Vote tallying happens in a central server in which the authorities have more control over the server. Integrating a distributed ledger could enhance the transparency and trust more for this kind of system rather than using a centralized mechanism. In Switzerland, some states using electronic voting system which has paper-based backups for security. Australia also has run trials in some states but large-scale adoption is not yet done.

2.2 Researches on electronic voting systems and related technologies

There are different research publications available about electronic voting solutions with various technologies to solve the challenges of adopting an electronic voting system. The proposed systems can be divided mainly into two groups as centralized and distributed. Mohammad et al. proposed a system to solve the cost effectiveness, accuracy and transparency problems and the proposed system includes conducting a paper based manual audit of the final report which is printed by the voting machine by every representative per polling station before sending it to the district's committee (Mohammad Hosam Sedky and Ramzy Hamed, 2015). Proposed system has its own database for each pooling station. The proposed system does not provide the transparency for the voters and it is much similar to the manual process. Küsters et al. proposes a tally hiding mechanism with end-to-end verifiable voting process (Küsters et al., 2020). Author identifies that in most of the elections, there will be a single winner or selecting the first few candidates based on the rankings so that it is not needed to reveal the complete tally. In this system, voters use public key and homotopic encryptions to encrypt and send their vote. In the tally phase, the votes will be calculated without decrypting vote. But the proposed model is not a scalable approach with its built-in complexity to verify the vote and auditing.

Blockchain technology which was introduced by Nakamoto has been identified as a better technology by many research's and articles (Nakamoto, 2008). Blockchain is used mainly in crypto currencies as a peer-to-peer distributed ledger and it has been evolving rapidly in crypto as well as in various industries like supply chain and health records management. Blockchain uses consensus mechanism (proof of work) to approve the transaction blocks by the nodes of the network. The process is called as mining where each transaction block needs to be accepted by solving a solving a cryptographic puzzle based on hashing. Blockchain achieves transparency and tamper avoidance nature with distributed ledger architecture and it consists of a chain of blocks, where each block contains a list of transactions and these blocks are cryptographically linked together, forming an immutable and tamper-proof record of data. Researchers have explored the possibility of adopting blockchain technology for the electoral voting process because of its potential which could change the way of an ordinary online voting system. Blockchain technology can be divided into permission less blockchain and permissioned blockchain based on the accessibility ("Permissioned and Permissionless Blockchains," 2019).

Yacoubi et al. proposes an algorithm for a electronic voting machine (EVM) which can protect the voter anonymity while also verifying the outcome in real time (Yacoubi et al., 2021). The proposed EVM uses the user id and the fingerprint for authentication with IOT devices and it uses the blockchain technology to ensure transparency in a decentralized manner. The algorithm uses a private key which is only user known and use that to generate a hash (private key + nonce). In the blockchain ledger the hash can be considered as the voter which could be used to verify and audit the casted vote by the voter. Author says this approach could increase the security and transparency while reducing ballot staffing, election cost and political confusion. The proposed system uses a database to store the voter ids and fingerprints casted which is used to stop the double spending or the ballot staffing. Using a non-distributed database is not the best approach for this type of systems since there is a possibility of the data tampering. The scalability improvements of the algorithm should be researched for a large-scale network.

Xiao et al. conducted a comprehensive survey on blockchain based voting systems and the survey says the central database could result in the problems such as lack of transparency, making it easier to be tampered and forged. Also, voters cannot verify the voting results and with blockchain will allow making the voting process open and transparent, preventing fraudulent votes, enhancing the security of voting data and verifying the voting results (Xiao

et al., 2020). But it also mentions that blockchain with a large network might need to consider how to improve the transaction throughput since large-scale voting scenarios require high timeliness and throughput. Author says that to improve transaction throughput, optimizations are needed in algorithm, block size, block generation time and transaction verification time.
Jafar et al. provides a critical review of blockchain-based e-voting systems, emphasizing scalability issues and potential security risks (Jafar et al., 2021). It says that the scalability issues also can be seen in blockchain based systems. The paper suggests using parallelization to mitigate the scalability issues which is called sharding. It also concludes that adopting blockchain voting methods might expose the voters to unforeseen security risks and flaws so that security concerns also should be investigated.

Anwar ul Hassan et al. points out the strengths of using blockchain for developing a liquid democracy enabled voting system (Anwar ul Hassan et al., 2022). The paper shows that blockchain's distributed and immutable transaction ledger increases the transparency and availability since distributed ledger will not have a single point of failure cases. In this paper, author proposes a system architecture based on one of the permissioned blockchain framework called Hyperledger Fabric framework to address flaws and ensure security, transparency and anonymity of the voter. The author says that the permissioned blockchain can enable proof of authority (POA) where the authorized nodes can verify the nodes in the network and predetermined boundaries can be enforced. Voters will get the transaction id after casting their vote and they can verify using the ledger that their vote is calculated but voters will not be able to check whether their vote is calculated is same as the choice submitted. The paper also mentions issues related to public scalability and processing overhead which should be addressed in future.

Majumder et al. identifies different consensus mechanisms used in blockchain based applications. The author proposes a system which implemented using "Exonum" which is a permissioned blockchain framework with hybrid consensus mechanism (Majumder et al., 2024). But the proposed system fails to achieve verifiability of the votes by voters. Jafar et al. has conducted a systematic review on blockchain based electronic voting systems and the author identifies drawbacks in existing systems such as coercion resistance, receipt freeness and scalability and performance issues associated with public blockchains (Jafar et al., 2022).
Taş et al. proposes a homophobic encryption-based manipulation prevention model for blockchain based e voting systems (Taş and Tanrıöver, 2021). According to the author, the proposed homophobic encryption method will enable the vote calculation without decrypting

the actual value which protects the voter anonymity and privacy with coercion resistance. But the propose method does not offer verifiability and transparency in counting votes. Chowdhury et al. has conducted an analysis of the distributed ledger platforms including Ethereum, Hyperledger Fabric, R3 Corda and some other frameworks (Chowdhury et al., 2019). Author identifies Corda framework is more suitable for financial purposes and has higher trust level compared to other frameworks.

Consensus can be identified as a term for a general agreement between the peers of the network. The purpose of the consensus is mutual acceptance of the validity of the transactions or data which is shared between nodes.
Bitcoin and Ethereum uses Proof-of-work as their consensus. The miners will try to solve the hashing puzzle and the miner which solves the puzzle first will get the chance to add the next block to the ledger. They will get rewarded with the crypto coins and the block will be accepted. But this is considered as more power inefficient and not an environment friendly process. Proof-of-stake and delegated-stake are considered more efficient and environment friendly than proof-of work since those are using betting mechanism with contributing a value as collateral. In non-competitive consensus, nodes in the network are trusted so that nodes can agree upon things instantly without delay as in competitive consensus. So, the non-competitive consensus can solve the performance issues and uncertainty issues.

The reviewed literature showcases the potential of using electronic voting systems to address flaws in paper based manual voting processes and provides evidence of the viability of using blockchain distributed ledger in electronic voting systems. It also shows that there are lack of transparency, auditability and performance related concerns of the proposed solutions. The reviewed literature shows that competitive consensus might have performance issues and delays of verifications while non-competitive methods can achieve instant verifications and reduce transaction time.

It identifies key areas of focus for the proposed project, such as transparency, auditability, scalability to contribute to the advancement of secure and transparent e-voting solutions.

CHAPTER 3

METHODOLOGY

This study will be using Design Science Research (DSR) approach to design, develop and evaluate the blockchain based electronic voting system. The DSR approach is particularly well-suited for addressing complex problems and developing innovative solutions and its iterative approach will align well with the practical nature of building a prototype of the proposing system. This study will mainly focus on areas of transparency, auditability, scalability and data transmission security of the system. The system will be designed for a hybrid environment where voting will be done in a controlled environment and vote verification and auditing can be done in a hybrid manner.

3.1 Selected technologies

The proposed system uses java as the core language and spring boot for implementing services. For this study, Corda framework ("Corda Community Edition Key Concepts - Community & Open Source 4.11," 2024) is used as the permissioned distributed ledger. Corda framework is a widely recognized private decentralized blockchain network platform which was initially developed to address the challenges in financial services but Corda 's key features, such as its focus on privacy, security, and interoperability, make it suitable for a wide range of applications beyond finance. Vaadin framework, which is a recognized server-side rendering framework is used for creating UI components.

3.2 Key concepts and terms

3.2.1 Distributed Ledger

Distributed ledger (DLT) is a database of records or transactions which is shared, replicated, and synchronized among the nodes of the network. DLTs allows participants of the network to have the consistency and transparency of the recorded data by restricting the data rigging. Each participant in the network has its own copy of the ledger, and changes to the ledger are independently and collectively validated by the participants through consensus mechanisms. The proposed system uses the Corda DLT framework to distribute the data among nodes. Corda has a vault service which stores the data of the ledger.

3.2.2 Corda network

Corda network is a permissioned network and only authorized nodes will be able to join the network. Each node has assigned a unique certificate and a public key by the network

operator. Each node has a mapping of a unique IP address which will be used in peer to peer communications by the network.

3.2.3 Consensus mechanism

Consensus mechanisms can be identified as a common understanding of how to ensure the validity of the records or transactions within the network. There are many different consensus algorithms used by different distributed systems. Widely recognized mechanisms include proof-of-work, proof-of-stake, proof-of-authority, byzantine fault tolerance etc. The consensus mechanism used in this study is Corda notaries and smart contracts. Notaries are special nodes which helps to maintain the validity and uniqueness of the transactions and prevent double spending. Corda smart contracts ensure the validity consensus of a particular transaction state.

3.2.4 States and Flows

States are immutable objects which contains the ledger data where each state maintains a unique identifier when creating the state. States are stored inside the vaults of the nodes. A flow is a sequence of steps which has the instructions on how to create or update the specific state inside the ledger. Corda flow framework maintains the updates of the states with all the participants of the transaction. The basic flow sequence is shown as below in figure 1.

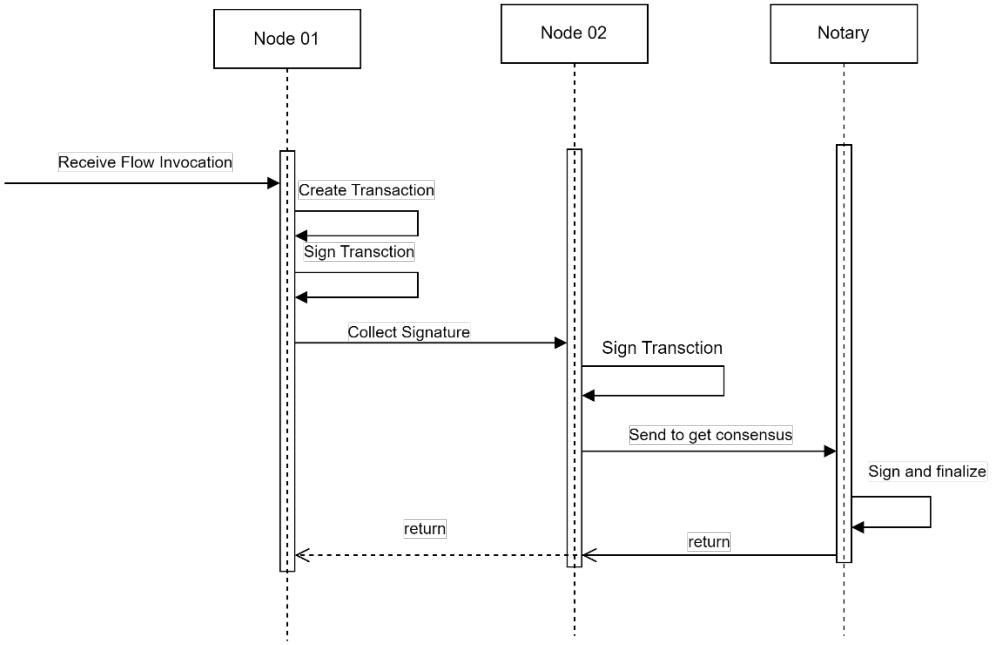


Figure 1: Corda flow sequence

3.3 System functionalities

Functionalities of the proposing system includes voter registration, candidate registration, voting and vote counting, vote verification by voter. In this study the main focus area is the voting, tallying and vote verification by the voter. Voter registration, Candidate registration and Political Party registration processes are done by the authorized system administrator.

3.3.1 Voting flow

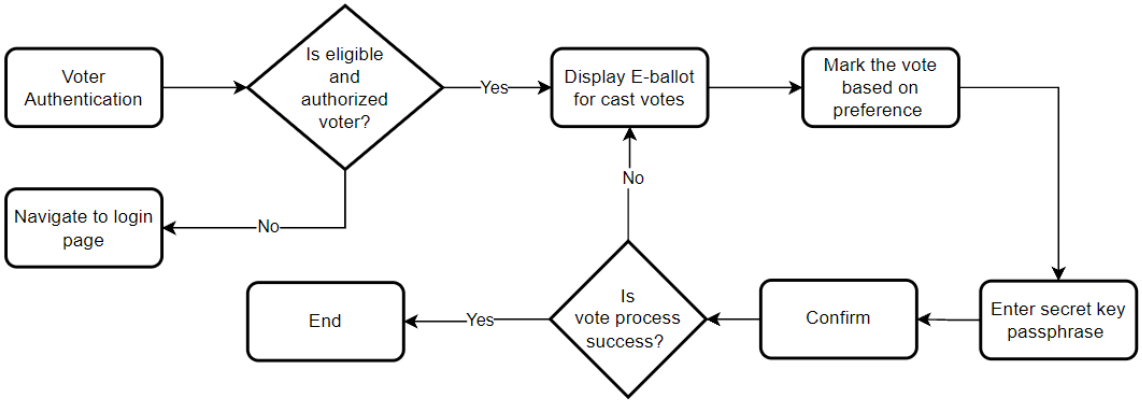


Figure 2: Voting flow

Voters need to go to the voting centers to cast their vote. Voter accounts are already created in voter registration which is handled separately before the election date. Voter will be logged in to the system using the credentials. After successful authentication, electronic ballot paper will be displayed with candidate and political party information. Voter can mark the choice and confirm. In the confirmation window, there will be a passphrase. This passphrase is used to encrypt the generated private key which is used encrypt the actual vote reference and choice. Voter needs to remember this passphrase, otherwise voter will not be able to verify the casted vote. After the confirmation, the vote will be submitted to the system and system issues the ballot paper reference receipt.

3.3.2 Vote information encryption

As shown in Figure 3, Vote information encryption is done when voter submitted the vote. The process creates two objects, Results Counting Object (Ob1) and Vote reference Object (Ob2). Ob1 contains the candidate reference. Keypair are generated as Public and Private and Private Key is encrypted using the passphrase provided at the vote submission by the voter. Ob2 contains the vote information encrypted with generated Public Key, Encrypted Private Key and the encrypted reference for the Ob1.

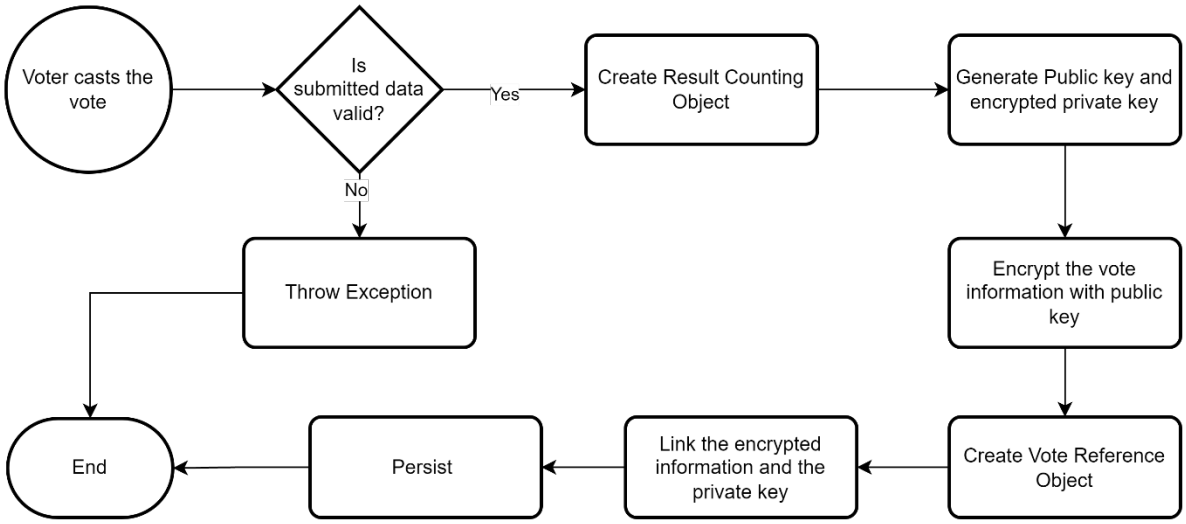


Figure 3: Encryption of vote information

3.3.3 Post-vote verification flow

Voter is able to verify their vote submitted to the system using the reference received after casting the vote and the passphrase given at the submission of the vote. For the verification, voter can use the voting machine allocated in voting center or can access the public verification endpoint with their own computer or smartphone. Voter only can verify their own submission.

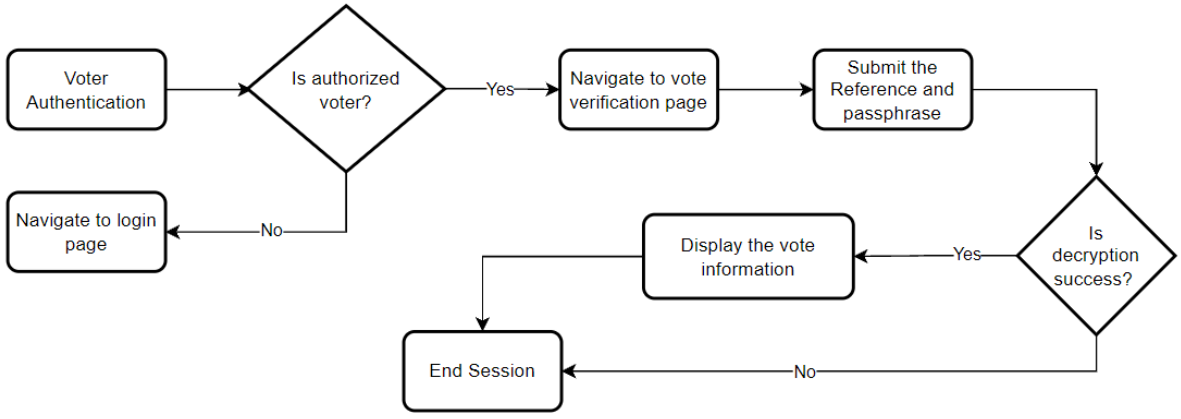


Figure 4: Post-vote verification flow

3.3.4 Vote information decryption

As in Figure 5, Information decryption is done in post-vote verification phase. Ob2 referred in is retrieved using the reference provided by the voter. Private Key attached to Ob2 is decrypted using the passphrase provided in vote submission. Using the decrypted Private Key,

the submitted vote information and the actual result counting object data will be decrypted and returned for the voter.

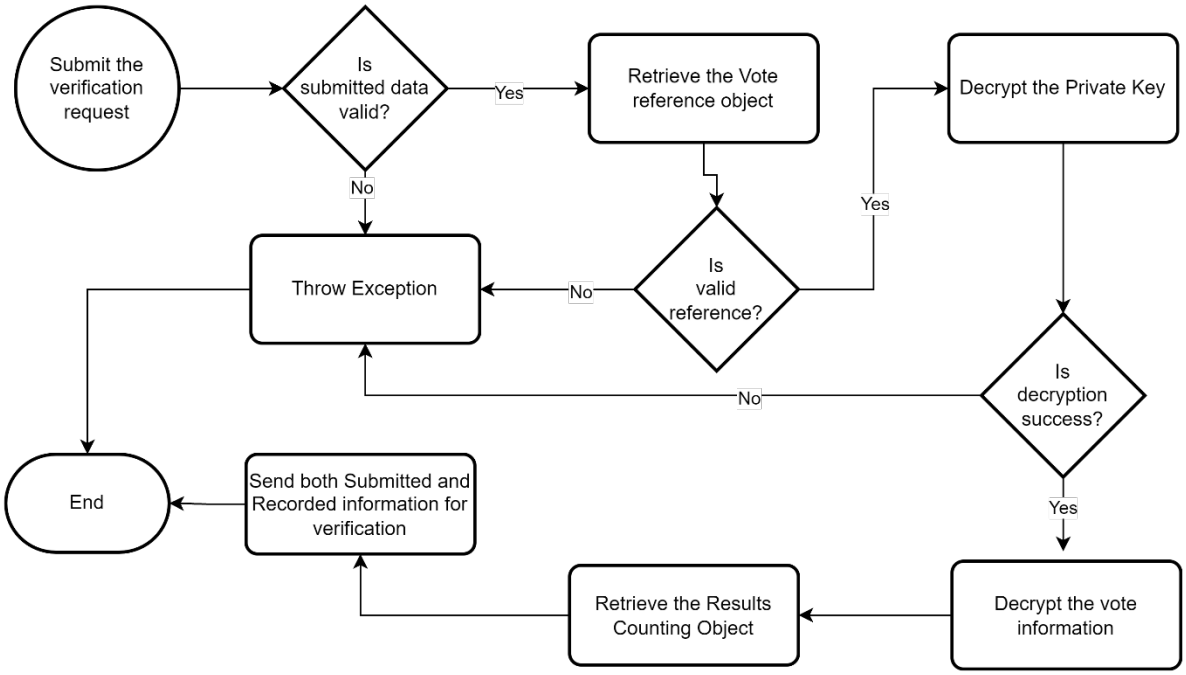


Figure 5: Vote information decryption

3.3.5 System generated keys and purposes

Key	**Type**	**Purpose**
Public Key	java.security.PublicKey	This key is used to encrypt the vote and voter information in voting phase.
Private Key	java.security.PrivateKey	This key is used to decrypt the ballot information in post voter verification phase.
Passphrase	String	This is used to encrypt and decrypt the private key for secure storage and to block the voter information decryption the third parties other that voter. Passphrase needs to be remembered by the voter in order to perform the post voter verification.

Table 1: System generated keys and purposes

3.4 Design

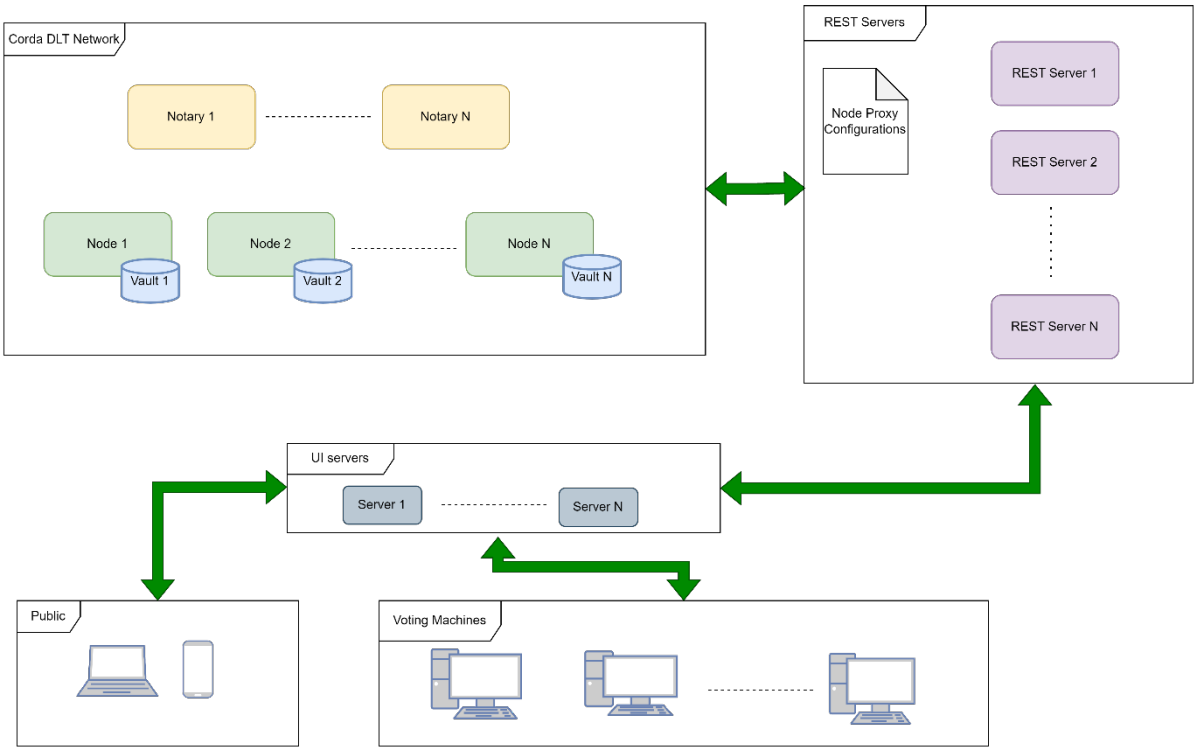


Figure 6: Proposed system architecture

The proposed system uses the decentralized blockchain ledger to achieve transparency and immutability of transactions using a peer to peer network architecture within the blockchain network. System consists of mainly three components which are User interface component, Rest API component and Corda DLT network component as in figure 3. Communication protocol used to call the REST endpoints is Hypertext Transfer Protocol Secure (HTTPS) and Internal services of the REST server uses Corda RPC protocol to communicate with the Corda network and within the nodes. Corda RPC protocol internally uses Advanced Message Queuing Protocol (AMQP) as the underlying messaging protocol. Proposed system uses JSON Web Tokens to share the claims between UI and REST Layers.

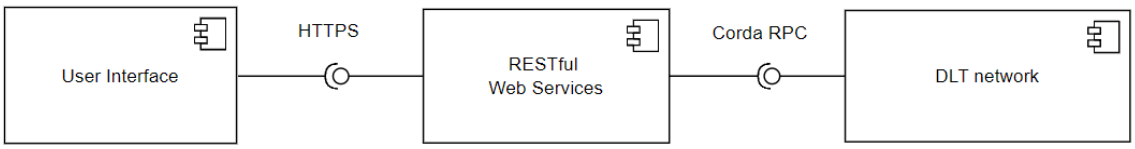


Figure 7: System Components

3.4.1 User Interface Component

User Interfaces are needed for authorized users such as administrators, voters to be interact with the system. Administrators have the permission to configure election meta data such as election type, time period, provinces, electoral districts, pooling divisions and voting centers, political parties, candidates and voters. Voters will interact with the UI for the voting and vote verification phases. Also, there are public interfaces for viewing the results and current status of the election. As I mentioned earlier, Vaadin framework is used to develop these views.

3.4.2 Rest API Component

The purpose of the REST API layer is to act as a middleware between UI and Distributed Ledger network. It helps to create a secure way of communication with nodes without exposing the node endpoints to outsiders and add additional protection. Using REST also enables the compatibility in different frameworks.

3.4.3 DLT Network Component

Distributed Ledger Network consists of mainly two types of nodes as Notaries and Registries. Since we need to share all the states between the registries to enable the transparency, pre-configured authorized nodes will be required in order to see the information shared with the network. Nodes can have multiple user types with credentials to allow different types of actions including, Read, Write, Sign, Start Flows. Internal services of the REST APIs need to connect with correct credentials to perform an action against the network.

3.5 Implementation

Implementation of corda application consist of several core implementations. They are State, Contract and Flow.

3.5.1. States

State is an object which used in storing data inside the node's vault. Corda transactions are based on the creation and update of states based on different entities. Each entity has its own data object. The list of implemented states are as follows.

State	**Description**
ElectionState	Stores election type, timeframe and status data.
PoliticalPartyState	Stores political party identity, name and registered election reference.

CandidateState	Stores candidate identity, name, political party reference and other registration details.
VoterState	Stores voter identity, name and other registration details
BallotState	Stores the encrypted vote information.
CandidateVoteState	Represents individual vote for a candidate. Stores candidate reference of the vote.
ProvinceState	Stores province details.
ElectoralDistrictState	Stores electoral district details with owning province reference.
PoolingDivisionState	Stores pooling division information with owning electoral district reference.

Table 2: States

A state must have implemented the required Corda State interfaces in order to store inside the ledger and vault. The implemented States has a BaseState class created with implementing the common logic related to the Corda states and it accepts the BaseDTO which holds the data object in the state. State's DTO is extended with BaseDTO.

```
// BaseState.java
public abstract class BaseState<D extends BaseDTO> implements
QueryableState, LinearState {…

// BaseDTO.java
@CordaSerializable
public abstract class BaseDTO implements Serializable {…
```

Figure 8: State and DTO common logic implementation example

3.5.2 Contracts

Contracts defines the rules of states transition from one valid state to another within the Corda network. Contracts in Corda are expressed as smart contracts, which ensure the integrity and validity of transactions on the distributed ledger. Every State is mapped to a corresponding Contract. Every contract is implemented the "net.corda.core.contracts.Contract" interface and needs to have a unique ID. Contact and State linking is done using annotation "net.corda.core.contracts.BelongsToContract" as in Figure 9.

```
// BallotContract.java
public class BallotContract implements Contract {…

    public static final String ID =
"org.chainballotx.contracts.BallotContract";

// BallotState.java
@BelongsToContract(BallotContract.class)
```

```
public class BallotState extends BaseState<BallotDTO> {…
```

Figure 9: Contract and State linking

Each Contract has its own command set defined by extending the "net.corda.core.contracts.CommandData" interface in order to use it in the Corda flow transactions.

3.5.3 Flows

Corda flows represent the sequence of steps that nodes follow to reach an agreement on creating or updating a state. Flows enable the execution of complex business processes in a distributed and secure manner. Flows in Corda are defined as classes that implement the "FlowLogic" interface. These classes contain the logic for the steps involved in a particular business process, such as initiating a transaction, collecting signatures, and reaching consensus with other nodes.

The list of implemented flows are as follows.

Flow	**Description**
ElectionCreationFlow	Initiates and responds to election state changes.
ProvinceCreationFlow	Initiates and responds to province state changes.
ElectoralDistrictCreationFlow	Initiates and responds to electoral district state changes.
PoolingDivisionCreationFlow	Initiates and responds to pooling division state changes.
PoliticalPartyRegistrationFlow	Initiates and responds to political party state changes.
CandidateRegistrationFlow	Initiates and responds to candidate state creations.
CandidateUpdateFlow	Initiates and responds to candidate state updates.
VoterRegistrationFlow	Initiates and responds to voter state creations.
VoterUpdateFlow	Initiates and responds to voter state updates.
BallotCastFlow	Initiates and responds to ballot cast state changes.

Table 3: Flows

Each Corda Flow will have a Initiator and Responder which returns Custom Data Object called FlowResult which contains the results that needs to be returned. Core implementations related to a flow as in Figure 10.

```
// BallotCastFlow.java
public class BallotCastFlow {…

@InitiatingFlow
@StartableByRPC
```

```
public static class Initiator extends BaseCreateFlowInitiator<BallotDTO,
BallotState> {…}

@InitiatedBy(BallotCastFlow.Initiator.class)
    public static class Responder extends BaseCreateFlowResponder {…}

// BaseCreateFlowInitiator.java
public abstract class BaseCreateFlowInitiator<D extends BaseDTO, S extends
BaseState<D>> extends FlowLogic<FlowResult> {…}

// BaseCreateFlowResponder.java
public abstract class BaseCreateFlowResponder extends FlowLogic<FlowResult>
{…}
```

Figure 10: Flow implementation example

3.5.4 REST API implementation

REST APIs are implemented using the Spring boot Rest Controllers. The implemented endpoints are as follows.

Endpoint	**Request Type**	**Permission**	**Purpose**
/api/v1/admin/voters	GET	ADMIN	Get the voter information
/api/v1/admin/voters/create	POST	ADMIN	Register new voter.
/api/v1/admin/voters/update	POST	ADMIN	Update existing voter information.
/api/v1/admin/candidates	GET	ADMIN	Get the candidate information.
/api/v1/admin/candidates/create	POST	ADMIN	Register new candidate.
/api/v1/admin/candidates/update	POST	ADMIN	Update existing candidate information.
/api/v1/admin/elections	GET	ADMIN	Get the election information
/api/v1/admin/elections/create	POST	ADMIN	Configure new election
/api/v1/admin/elections/update	POST	ADMIN	Update the configured election.
/api/v1/admin/political-parties	GET	ADMIN	Get the political party information.
/api/v1/admin/political-parties/create	POST	ADMIN	Register new political party.
/api/v1/admin/provinces	GET	ADMIN	Get the configured provinces.
/api/v1/admin/provinces/create	POST	ADMIN	Configure province.

/api/v1/admin/electoral-districts	GET	ADMIN	Get the configured electoral districts
/api/v1/admin/electoral-districts/create	POST	ADMIN	Configure electoral districts.
/api/v1/admin/pooling-divisions	GET	ADMIN	Get the configured pooling divisions.
/api/v1/admin/pooling-divisions/create	POST	ADMIN	Configure pooling divisions.
/api/v1/ballots/cast	POST	VOTER	Cast the vote.
/api/v1/ballots/verify	POST	VOTER	Verify the vote information.
/api/v1/results	GET	PUBLIC	Get the results information of the election.
/api/v1/auth/login	POST	PUBLIC	Returns the JWT claims based on authentication information.

Table 4: REST API endpoints

Jason Web Tokens (JWT) are used to access and share the claims of the user. JWT created is signed using the secret key configured in the API server and it is used to verify that the sender of the JWT is who it says it is and to ensure that the message wasn't changed along the way.

JWT generator function is as follows.

```
public String generateToken(UserDetails userDetails) {
    List<String> roles = userDetails.getAuthorities().stream()
            .map(GrantedAuthority::getAuthority)
            .collect(Collectors.toList());

    return Jwts.builder()
            .setSubject(userDetails.getUsername())
            .claim("roles", roles)
            .setIssuedAt(new Date())
            .setExpiration(new Date(System.currentTimeMillis() +
validityInMilliseconds))
            .signWith(SignatureAlgorithm.HS512, secretKey)
            .compact();
}
```

3.6 Build tools and deployment

Build tools used in development are Maven and Gradle.

Containerized application deployment is used to deploy the UI server, REST API server and Corda Nodes using Docker images.

3.7 System prototype

The developed system prototype has UI server, REST API Server and Corda Network which includes two Peer nodes and one Notary node configured. Prototype is deployed in a single machine (PC) with simulation tools.

CHAPTER 4

EVALUATION AND RESULTS

4.1 Research hypotheses

As mentioned in above sections, this research project focuses on enhancing the transparency and security of voting and auditability in the electronic voting system by investigating the integration of Corda, a permissioned distributed ledger platform to address existing challenges and enhance the integrity of the electoral process.

Research hypotheses and questions are as follows

- How to ensure transparency of the e-voting process by integrating with a permissioned distributed ledger platform.
- How auditability and verifiability of the casted votes can be achieved in the proposed e-voting system.
- How does the e-voting system address privacy and anonymity features associated with voter information while maintaining the integrity of the electoral process?

4.2 Evaluation approach

This study uses the experiment-based evaluation approach to evaluate the system. It includes execution of controlled experiments to assess the transparency, auditability, and voter anonymity of the e-voting system.

4.2.1 Evaluation scope

For the simulations, scope of the election type will be Presidential Election and there will be pre-configured time frame before start each simulation.

4.2.2 Dataset

Dataset is generated simulation dataset to simulate various voting scenarios. It includes voter, candidate and system administrator profiles to represent the relevant users of the system. Election process configuration metadata is added to the system before each evaluation cycle. The required simulation pre-configurations are as in Table 4.

Type	**Description**	**Values**
Election Type	The election type used for simulation	Presidential Election
Date	Date of the election	To be set at each simulation
Start Time	Start time of the election	To be set at each simulation

End Time	End time of the election	To be set at each simulation
Provinces	The set of provinces	2
Electoral Districts	Electoral districts of each province	4 (2 for each province)
Pooling Divisions	Pooling divisions of each district	8 (2 for each district)
Voting Centers	Voting centers of each pooling division	16 (2 for each district)
Political Parties	Registered political parties for the election.	3
Candidates	Competing candidate details for the election	3
Voters	Registered voter accounts for the election	To be set at each simulation

Table 5: Simulation pre-configurations

Data generation functions are implemented to generate the username, password, NIC, chosen-candidate and secret key for the Private Key encryption to simulate the voting scenarios. Random voting center is assigned to each voter and candidate choice also randomized and included in the csv for simulation. Set of generated data is as in Figure 11.

	A	B	C	D
1	voting_center	poolingDivision	electoralDistrict	province
2	ElectoralDistrict01_PD_01_VC_01	ElectoralDistrict01_PD_01	ElectoralDistrict01	Province01
3	ElectoralDistrict01_PD_01_VC_02	ElectoralDistrict01_PD_01	ElectoralDistrict01	Province01
4	ElectoralDistrict01_PD_02_VC_01	ElectoralDistrict01_PD_02	ElectoralDistrict01	Province01
5	ElectoralDistrict01_PD_02_VC_02	ElectoralDistrict01_PD_02	ElectoralDistrict01	Province01
6	ElectoralDistrict02_PD_01_VC_01	ElectoralDistrict02_PD_01	ElectoralDistrict02	Province01
7	ElectoralDistrict02_PD_01_VC_02	ElectoralDistrict02_PD_01	ElectoralDistrict02	Province01
8	ElectoralDistrict02_PD_02_VC_01	ElectoralDistrict02_PD_02	ElectoralDistrict02	Province01
9	ElectoralDistrict02_PD_02_VC_02	ElectoralDistrict02_PD_02	ElectoralDistrict02	Province01
10	ElectoralDistrict03_PD_01_VC_01	ElectoralDistrict03_PD_01	ElectoralDistrict03	Province02
11	ElectoralDistrict03_PD_01_VC_02	ElectoralDistrict03_PD_01	ElectoralDistrict03	Province02
12	ElectoralDistrict03_PD_02_VC_01	ElectoralDistrict03_PD_02	ElectoralDistrict03	Province02
13	ElectoralDistrict03_PD_02_VC_02	ElectoralDistrict03_PD_02	ElectoralDistrict03	Province02
14	ElectoralDistrict04_PD_01_VC_01	ElectoralDistrict04_PD_01	ElectoralDistrict04	Province02
15	ElectoralDistrict04_PD_01_VC_02	ElectoralDistrict04_PD_01	ElectoralDistrict04	Province02
16	ElectoralDistrict04_PD_02_VC_01	ElectoralDistrict04_PD_02	ElectoralDistrict04	Province02
17	ElectoralDistrict04_PD_02_VC_02	ElectoralDistrict04_PD_02	ElectoralDistrict04	Province02

	A	B	C	D	E	F
1	username	nic	votingCenter	vote	password	secret
2	Voter_0	0000000000V	ElectoralDistrict01_PD_02_VC_02	200000000V	password_0	secret_0
3	Voter_1	0000000001V	ElectoralDistrict02_PD_01_VC_01	100000000V	password_1	secret_1
4	Voter_2	0000000002V	ElectoralDistrict03_PD_01_VC_02	100000000V	password_2	secret_2
5	Voter_3	0000000003V	ElectoralDistrict02_PD_01_VC_02	300000000V	password_3	secret_3
6	Voter_4	0000000004V	ElectoralDistrict04_PD_01_VC_01	300000000V	password_4	secret_4
7	Voter_5	0000000005V	ElectoralDistrict04_PD_02_VC_01	100000000V	password_5	secret_5
8	Voter_6	0000000006V	ElectoralDistrict02_PD_02_VC_02	200000000V	password_6	secret_6
9	Voter_7	0000000007V	ElectoralDistrict02_PD_02_VC_01	100000000V	password_7	secret_7
10	Voter_8	0000000008V	ElectoralDistrict03_PD_02_VC_01	100000000V	password_8	secret_8
11	Voter_9	0000000009V	ElectoralDistrict03_PD_02_VC_01	200000000V	password_9	secret_9
12	Voter_10	0000000010V	ElectoralDistrict04_PD_02_VC_02	100000000V	password_10	secret_10
13	Voter_11	0000000011V	ElectoralDistrict02_PD_02_VC_01	100000000V	password_11	secret_11
14	Voter_12	0000000012V	ElectoralDistrict04_PD_01_VC_02	200000000V	password_12	secret_12
15	Voter_13	0000000013V	ElectoralDistrict03_PD_01_VC_01	100000000V	password_13	secret_13
16	Voter_14	0000000014V	ElectoralDistrict02_PD_01_VC_02	100000000V	password_14	secret_14
17	Voter_15	0000000015V	ElectoralDistrict02_PD_02_VC_01	200000000V	password_15	secret_15

Figure 11: Sample of generated data

4.2.3 Tools

JMeter is used as the main simulation tool to invoke the rest APIs of the system with predefined parameters. JMeter is an opensource software tool designed for load testing functionalities of web applications. It provides ways to test the concurrent user scenarios and the results and the performance measures can be exported for comparison and assessments.

4.3 Experiments

The set of preconfigured voters will participate in a controlled election process simulation through the system for the experiments to check whether the system expectations are met. JMeter is configured to record responses with Listener nodes such as JSR223 Listener, Summary Report, Response Time Graph and View Results Tree.

4.3.1 Experiment 1 - Accuracy

In this experiment, system is tested to check the casted vote counts and recorded vote counts are matches for each candidate. The data configurations which are added before the simulation are as in Table 5.

Configuration	**Value**
Eligible Voters	100

Table 6: Experiment 1 configurations

In this scenario, JMeter is configured to have 100 threads with ramp-up time 1 sec and iteration count 1 to simulate the concurrent users. HTTP request is configured to invoke the */api/v1/ballots/cast* endpoint with generated data set. System is expected to record the votes as submitted and the responses are compared with the data set used in the experiment.

4.3.2 Experiment 2 - Verifiability

In this experiment, system is tested to check the submitted choice of candidate and the recorded choice which uses in the final results are matches for each voter. The data configurations which are added before the simulation are as in Table 6.

Configuration	**Value**
Eligible Voters	100

Table 7: Experiment 2 configurations

In this scenario, JMeter is configured to have 100 threads with ramp-up time 1 sec and iteration count 1 to simulate the concurrent users. First HTTP request is configured to invoke the */api/v1/ballots/cast* endpoint and the second HTTP request is configured to invoke */api/v1/ballots/verify* endpoint to verify the vote with generated data set. Also, JMeter is configured to give the summary of the verification request aiming to get a meaningful input for the analysis of verification process. System is expected to record the votes as submitted

and the voters should be able to verify the casted vote using their ballot reference and passphrase and expected to pass all the verifications.

4.3.3 Experiment 3 - Double spend resistance

In this experiment, system is tested to check the resistance for the multiple vote casting. System is expected to block all the multiple attempts. Simulation configurations are as follows in Table 7.

Configuration	**Value**
Eligible Voters	100
Multiple attempts	50

Table 8: Experiment 3 configurations

In this scenario, JMeter is configured to have 150 threads with ramp-up time 1 sec and iteration count 1 to simulate the concurrent users. HTTP request is configured to invoke the */api/v1/ballots/cast* endpoint with generated data set of 100 voters. System is expected to only record the valid votes of 100. Other 50 records should be failed.

4.3.4 Experiment 4 - Stability

In this experiment, system is tested to check the stability with the load. This experiment is conducted in multiple iterations as 100, 1000 and 2500 threads. Simulation configurations are as follows in Table 7.

Iteration	**Configuration**	**Value**
1	Eligible Voters	100
2	Eligible Voters	1000
3	Eligible Voters	2500

Table 9: Experiment 4 configurations

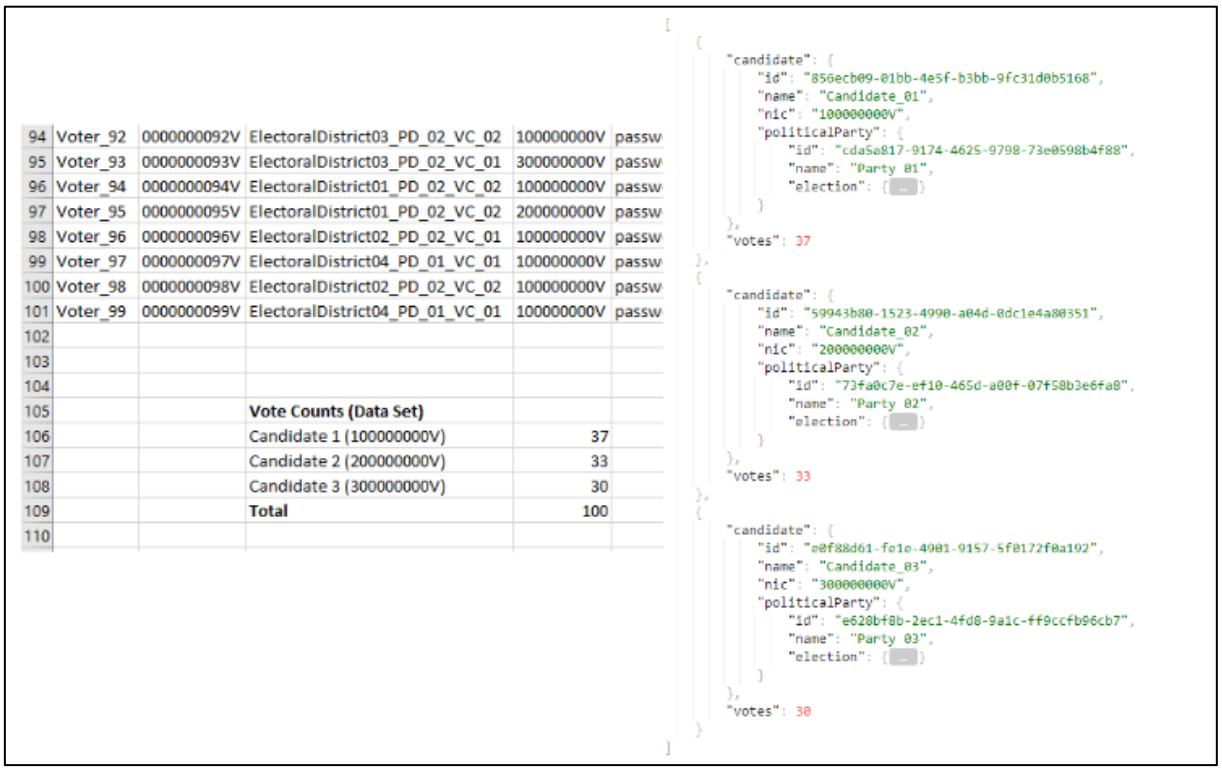
In this scenario, JMeter is configured to have respective thread counts in above table threads with ramp-up time 1 sec and iteration count 1 to simulate the concurrent users. HTTP request is configured to invoke the */api/v1/ballots/cast* endpoint with generated data set of voters. System is expected to process all votes without failures.

4.4 Simulation Results

This section describes the system evaluation results based on each controlled simulation conducted and the observations of the non-functional aspects of the system.

4.4.1 Experiment 1 - Accuracy Results

In experiment 1, system was able to record all the 100 submitted votes without error and the data set summary of each candidate vote counts and the system generated results are matched. The below figure shows the used Dataset summary and final results.



94	Voter_92	0000000092V	ElectoralDistrict03_PD_02_VC_02	100000000V	passw
95	Voter_93	0000000093V	ElectoralDistrict03_PD_02_VC_01	300000000V	passw
96	Voter_94	0000000094V	ElectoralDistrict01_PD_02_VC_02	100000000V	passw
97	Voter_95	0000000095V	ElectoralDistrict01_PD_02_VC_02	200000000V	passw
98	Voter_96	0000000096V	ElectoralDistrict02_PD_02_VC_01	100000000V	passw
99	Voter_97	0000000097V	ElectoralDistrict04_PD_01_VC_01	100000000V	passw
100	Voter_98	0000000098V	ElectoralDistrict02_PD_02_VC_02	100000000V	passw
101	Voter_99	0000000099V	ElectoralDistrict04_PD_01_VC_01	100000000V	passw
102					
103					
104					
105			**Vote Counts (Data Set)**		
106			Candidate 1 (100000000V)	37	
107			Candidate 2 (200000000V)	33	
108			Candidate 3 (300000000V)	30	
109			**Total**	100	
110					


```
[
  {
    "candidate": {
      "id": "856ecb09-01bb-4e5f-b3bb-9fc31d0b5168",
      "name": "Candidate_01",
      "nic": "100000000V",
      "politicalParty": {
        "id": "cda5a817-9174-4625-9798-73e0598b4f88",
        "name": "Party 01",
        "election": {...}
      }
    },
    "votes": 37
  },
  {
    "candidate": {
      "id": "59943b80-1523-4990-a84d-0dc1e4a80351",
      "name": "Candidate_02",
      "nic": "200000000V",
      "politicalParty": {
        "id": "73fa0c7e-ef10-465d-a00f-07f58b3e6fa8",
        "name": "Party 02",
        "election": {...}
      }
    },
    "votes": 33
  },
  {
    "candidate": {
      "id": "e0f88d61-fe1e-4001-9157-5f0172f0a192",
      "name": "Candidate_03",
      "nic": "300000000V",
      "politicalParty": {
        "id": "e620bf8b-2ec1-4fd8-9a1c-ff9ccfb96cb7",
        "name": "Party 03",
        "election": {...}
      }
    },
    "votes": 30
  }
]
```


Figure 12: Experiment 1 - Dataset summary and final results

Transaction throughput is recorded as 0.98/sec which is an acceptable value because of the used hardware resources. There were 100 threads in 1 second ramp-up period used in this simulation. It means that all 100 threads were initiated within the first second of the simulation. But the system was able to process the requests without failure confirming the ability to handle sudden surge in load. It also shows around 88 sec average latency which indicates that system is under heavy load.

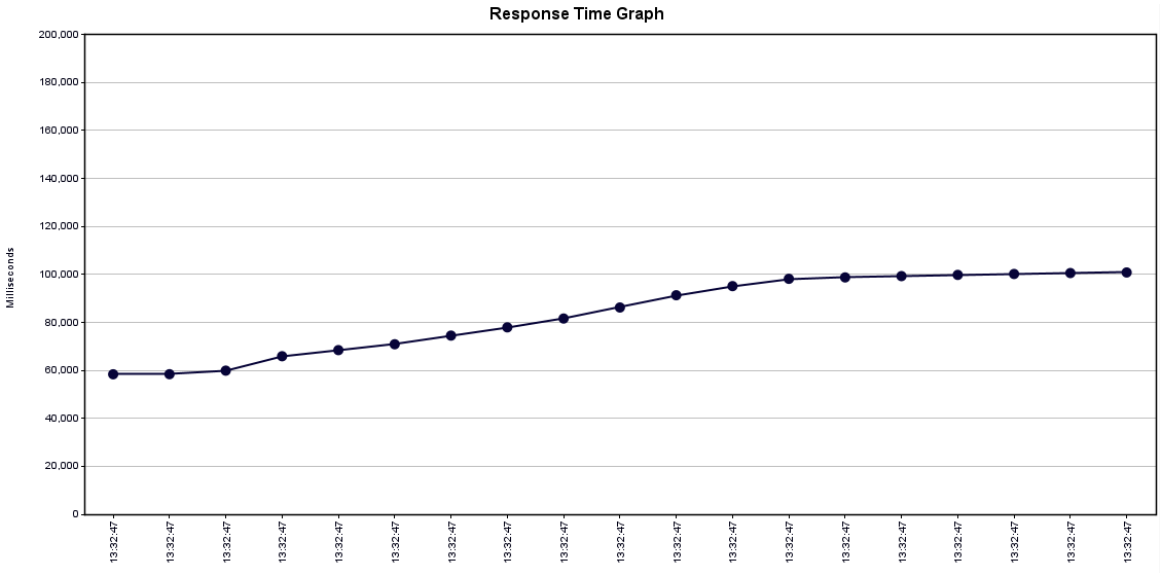


Figure 13: Experiment 1 - Response time deviation (interval 50 milliseconds)

The figure below shows the summary of the system performance.

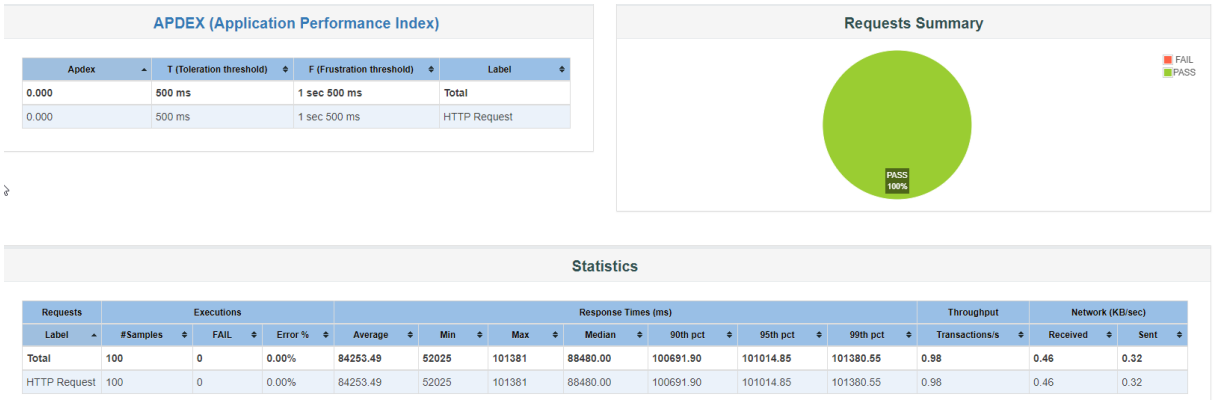
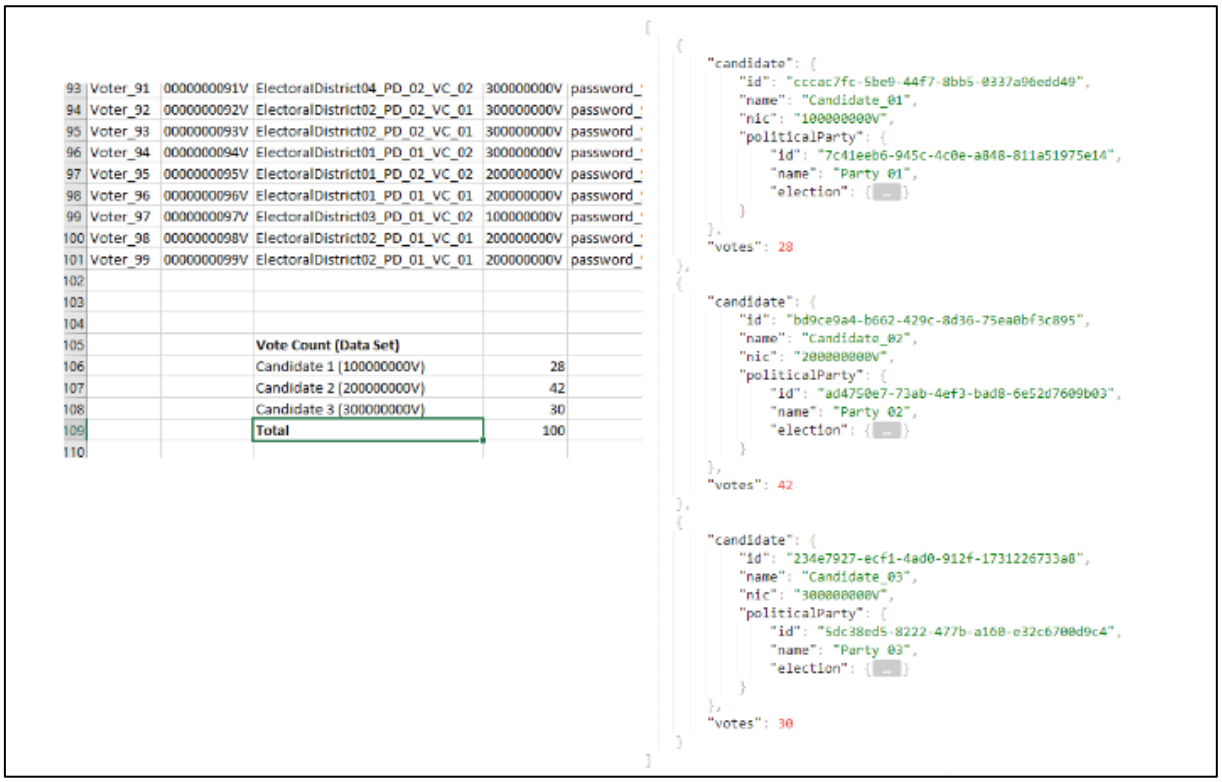


Figure 14: Experiment 1 - System performance summary

The simulation outcome proves that accuracy of the vote recording is as expected from the system and also the stability is observed through consistent response times and zero error rate.

4.4.2 Experiment 2 - Verifiability Results

In experiment 2, the total of 200 requests are generated with JMeter as 100 ballot casting requests and 100 verification requests. System was able to record all the 100 submitted votes and verified all the votes with voter secret keys successfully without any error and the data set summary of each candidate vote counts and the system generated results are matched. The below figure shows the used Dataset summary and final results.



93	Voter_91	0000000091V	ElectoralDistrict04_PD_02_VC_02	300000000V	password_
94	Voter_92	0000000092V	ElectoralDistrict02_PD_02_VC_01	300000000V	password_
95	Voter_93	0000000093V	ElectoralDistrict02_PD_02_VC_01	300000000V	password_
96	Voter_94	0000000094V	ElectoralDistrict01_PD_01_VC_02	300000000V	password_
97	Voter_95	0000000095V	ElectoralDistrict01_PD_02_VC_02	200000000V	password_
98	Voter_96	0000000096V	ElectoralDistrict01_PD_01_VC_01	200000000V	password_
99	Voter_97	0000000097V	ElectoralDistrict03_PD_01_VC_02	100000000V	password_
100	Voter_98	0000000098V	ElectoralDistrict02_PD_01_VC_01	200000000V	password_
101	Voter_99	0000000099V	ElectoralDistrict02_PD_01_VC_01	200000000V	password_
102					
103					
104					
105			Vote Count (Data Set)		
106			Candidate 1 (100000000V)	28	
107			Candidate 2 (200000000V)	42	
108			Candidate 3 (300000000V)	30	
109			Total	100	
110					


```
[
  {
    "candidate": {
      "id": "cccac7fc-5be9-44f7-8bb5-0337a96edd49",
      "name": "Candidate_01",
      "nic": "100000000V",
      "politicalParty": {
        "id": "7c41eeb6-945c-4c0e-a848-811a51975e14",
        "name": "Party 01",
        "election": {...}
      }
    },
    "votes": 28
  },
  {
    "candidate": {
      "id": "bd9ce9a4-b662-429c-8d36-75ea0bf3c895",
      "name": "Candidate_02",
      "nic": "200000000V",
      "politicalParty": {
        "id": "ad4750e7-73ab-4ef3-bad8-6e52d7609b03",
        "name": "Party 02",
        "election": {...}
      }
    },
    "votes": 42
  },
  {
    "candidate": {
      "id": "234e7927-ecf1-4ad0-912f-1731226733a8",
      "name": "Candidate_03",
      "nic": "300000000V",
      "politicalParty": {
        "id": "5dc38ed5-8222-477b-a160-e32c6700d9c4",
        "name": "Party 03",
        "election": {...}
      }
    },
    "votes": 30
  }
]
```


Figure 15: Experiment 2 - Dataset summary and final results

Transaction throughput is recorded as 2.06/sec which is a better value compared to previous experiment. This transaction time is recorded only for the verification process and the results confirms that ballot casting process is heavier than the verification process. The response deviation shows that the reduction of response time in the end of the simulation. Explanation of that is in the start of the simulation, ballot castings is also ongoing so that it has taken considerable amount of resources which causes verification delays in the beginning.

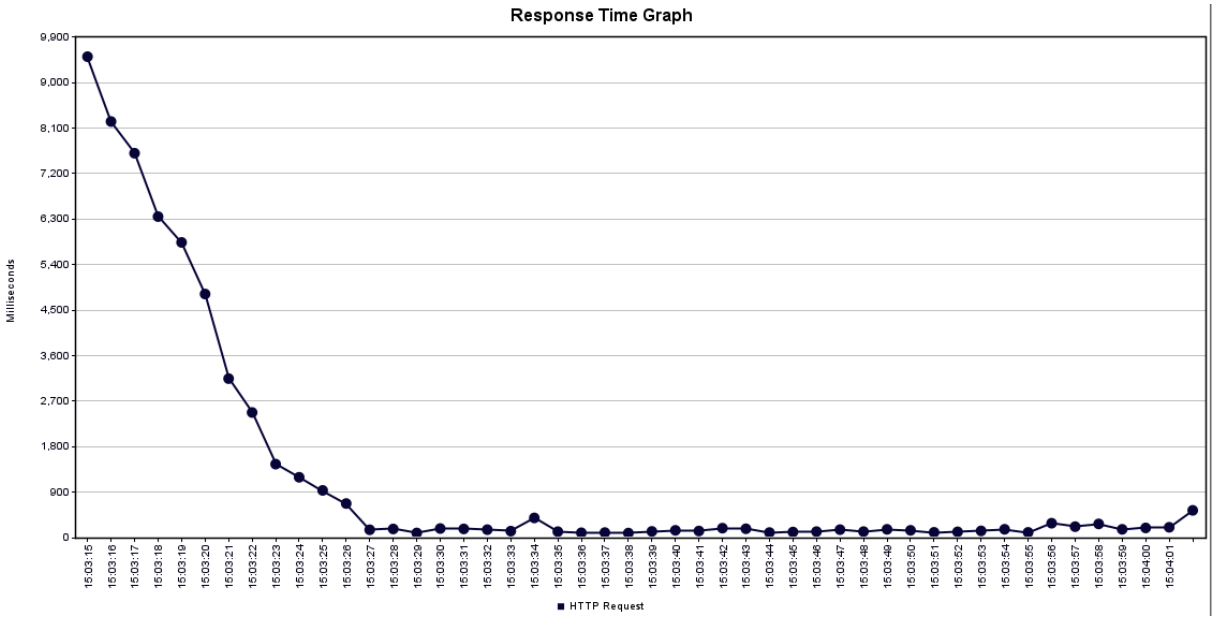


Figure 16: Experiment 2 - Response time deviation (interval 1 second)

The figure below shows the summary of the system performance of the simulation.

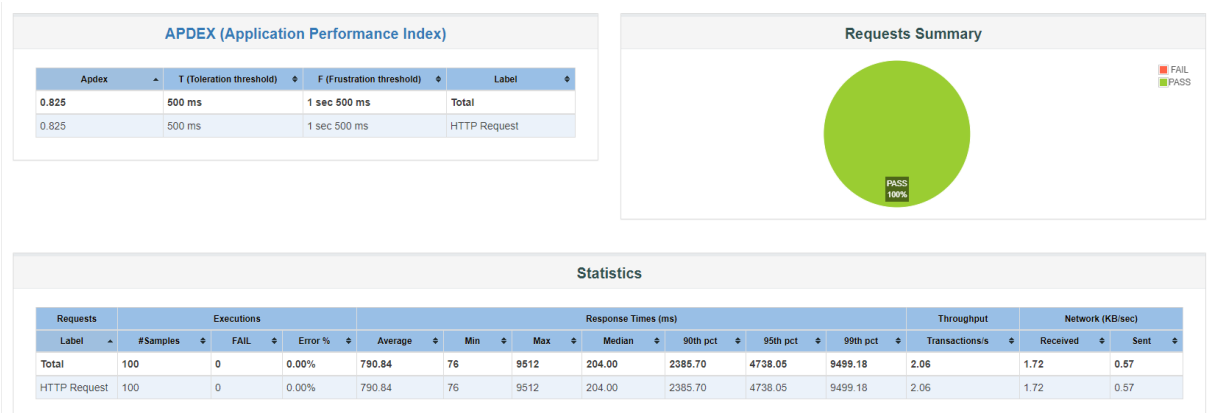
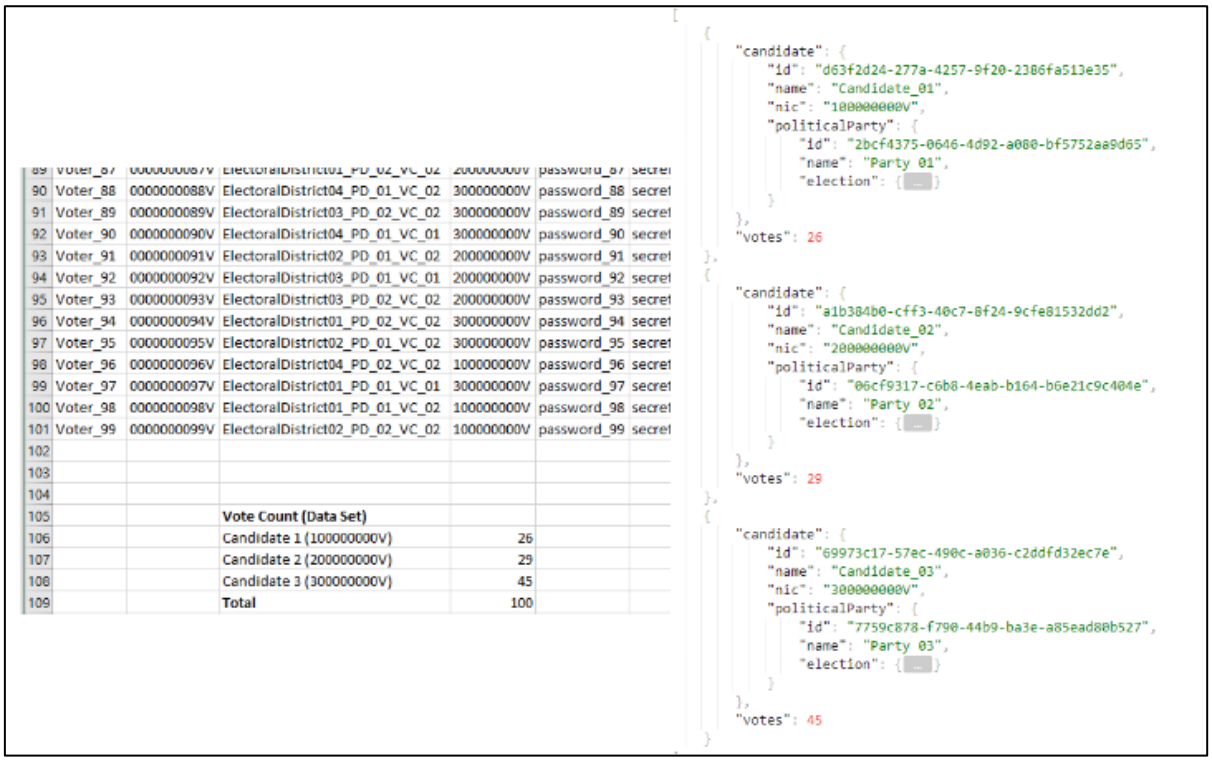


Figure 17: Experiment 2 - System performance summary

The simulation outcome proves that accuracy of the vote verification accuracy as well as the recording accuracy is as expected from the system and system is capable of verification of post votes are as expected. Also, the stability is observed through response times and zero error rate.

4.4.3 Experiment 3 - Double spend resistance

In experiment 3, ballot cast requests made using 150 threads and 50 out of the 150 is invalid and duplicate ones. System was able to block all the duplicate attempts made and successfully recorded all the valid votes.



90	Voter_88	0000000088V	ElectoralDistrict04_PD_01_VC_02	300000000V	password_88	secret
91	Voter_89	0000000089V	ElectoralDistrict03_PD_02_VC_02	300000000V	password_89	secret
92	Voter_90	0000000090V	ElectoralDistrict04_PD_01_VC_01	300000000V	password_90	secret
93	Voter_91	0000000091V	ElectoralDistrict02_PD_01_VC_02	200000000V	password_91	secret
94	Voter_92	0000000092V	ElectoralDistrict03_PD_01_VC_01	200000000V	password_92	secret
95	Voter_93	0000000093V	ElectoralDistrict03_PD_02_VC_02	200000000V	password_93	secret
96	Voter_94	0000000094V	ElectoralDistrict01_PD_02_VC_02	300000000V	password_94	secret
97	Voter_95	0000000095V	ElectoralDistrict02_PD_01_VC_02	300000000V	password_95	secret
98	Voter_96	0000000096V	ElectoralDistrict04_PD_02_VC_02	100000000V	password_96	secret
99	Voter_97	0000000097V	ElectoralDistrict01_PD_01_VC_01	300000000V	password_97	secret
100	Voter_98	0000000098V	ElectoralDistrict01_PD_01_VC_02	100000000V	password_98	secret
101	Voter_99	0000000099V	ElectoralDistrict02_PD_02_VC_02	100000000V	password_99	secret
102						
103						
104						
105			**Vote Count (Data Set)**			
106			Candidate 1 (100000000V)	26		
107			Candidate 2 (200000000V)	29		
108			Candidate 3 (300000000V)	45		
109			**Total**	100		


```
[
  {
    "candidate": {
      "id": "d63f2d24-277a-4257-9f20-2386fa513e35",
      "name": "Candidate_01",
      "nic": "100000000V",
      "politicalParty": {
        "id": "2bcf4375-0646-4d92-a080-bf5752aa9d65",
        "name": "Party 01",
        "election": {...}
      }
    },
    "votes": 26
  },
  {
    "candidate": {
      "id": "a1b384b0-cff3-40c7-8f24-9cfe81532dd2",
      "name": "Candidate_02",
      "nic": "200000000V",
      "politicalParty": {
        "id": "06cf9317-c6b8-4eab-b164-b6e21c9c404e",
        "name": "Party 02",
        "election": {...}
      }
    },
    "votes": 29
  },
  {
    "candidate": {
      "id": "69973c17-57ec-490c-a036-c2ddfd32ec7e",
      "name": "Candidate_03",
      "nic": "300000000V",
      "politicalParty": {
        "id": "7759c878-f790-44b9-ba3e-a85ead80b527",
        "name": "Party 03",
        "election": {...}
      }
    },
    "votes": 45
  }
]
```


Figure 18: Experiment 3 - Dataset summary and final results

It is observed that average response time is around 60 seconds and response deviation is as in figure below.

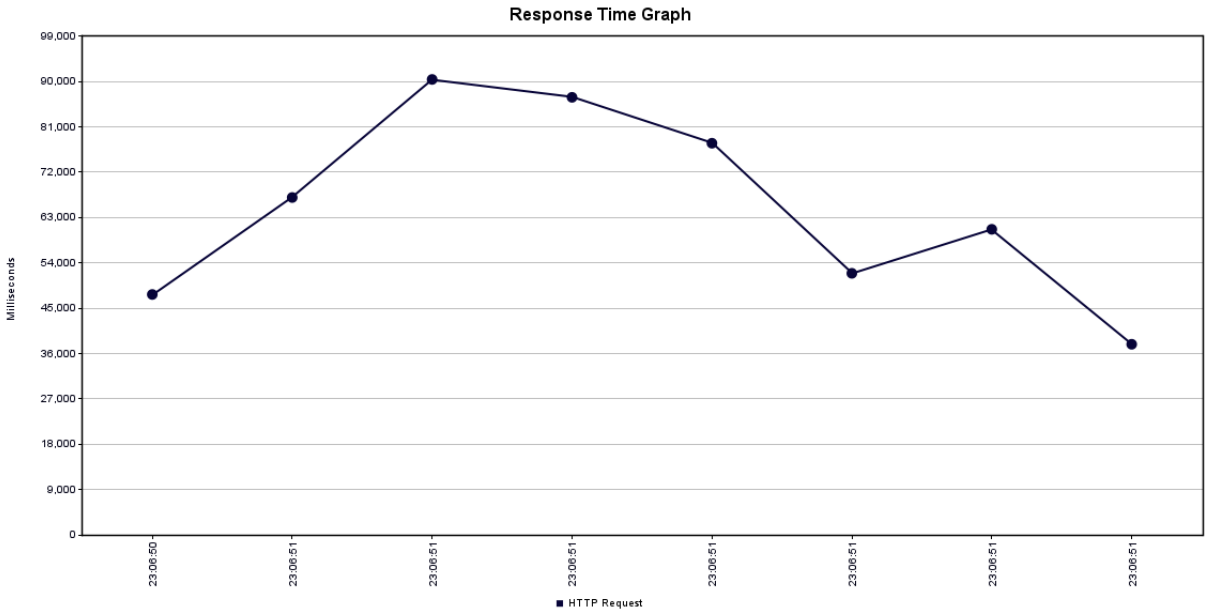


Figure 19: Experiment 3 - Response time deviation (interval 100 milliseconds)

Transaction throughput is recorded as 1.39/sec which is a higher value compared to the experiment 1 results. Simulation outcome shows that system is capable of handling multiple

vote attempts and double spent resistance. The below figure shows the summary of the system performance.

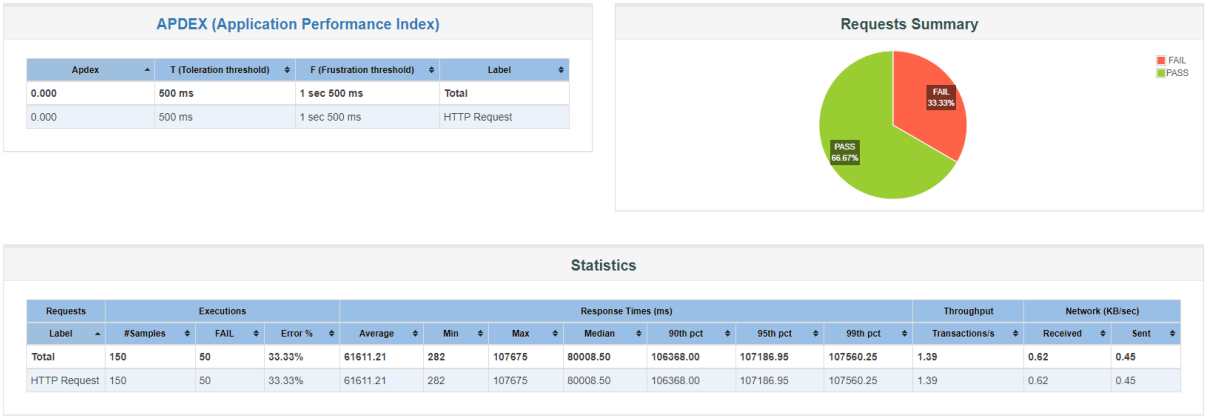


Figure 20: Experiment 3 - System performance summary

4.4.4 Experiment 4 – Stability

In each three iterations with load 100, 1000 and 2500, system was able to record all the requests successfully with error rate of 0%. In each three iterations the transaction times recorded as 1.05/s, 0.98/s and 0.93/s. The system performance summaries in each iteration are as in below three figures.

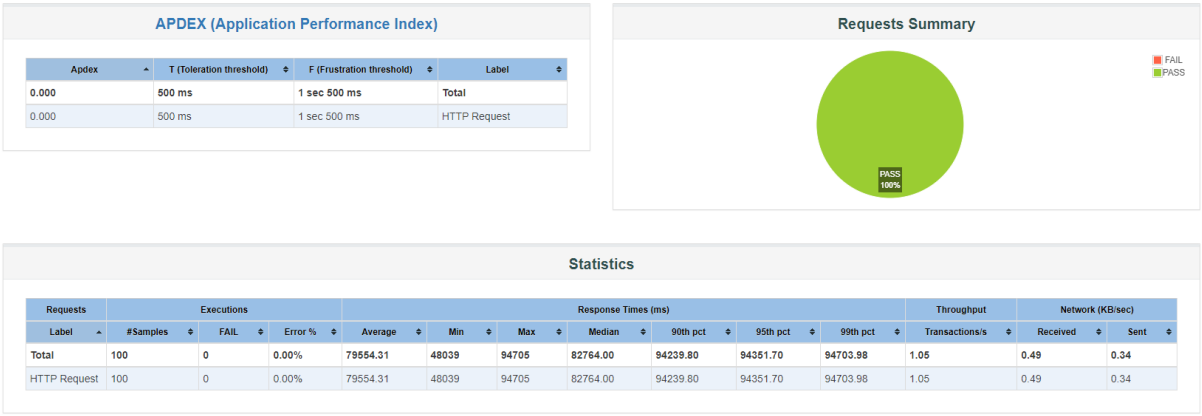


Figure 21: Experiment 4 Iteration 1 – System Performance

APDEX (Application Performance Index)

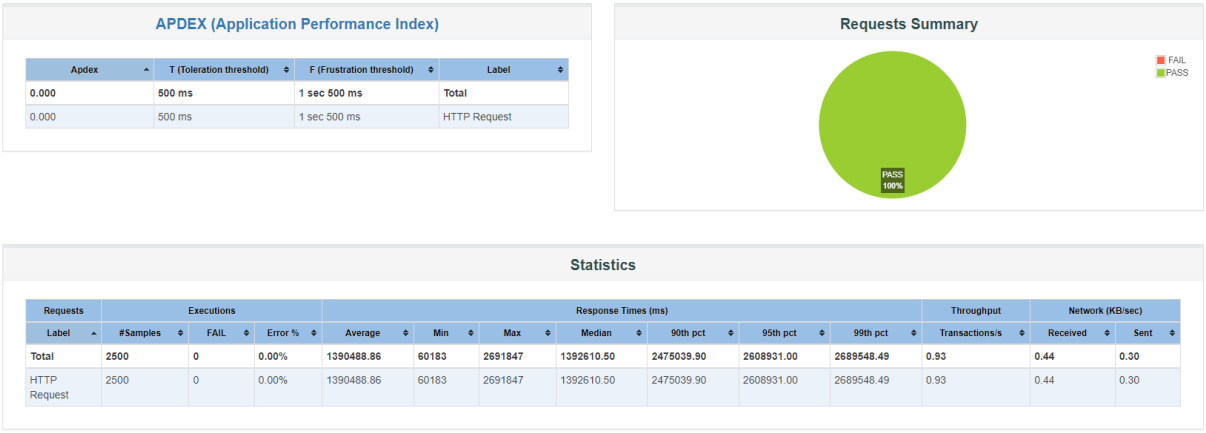
Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.000	500 ms	1 sec 500 ms	Total
0.000	500 ms	1 sec 500 ms	HTTP Request

Requests Summary

Statistics

Requests	Executions			Response Times (ms)							Throughput	Network (KB/sec)	
Label	#Samples	FAIL	Error %	Average	Min	Max	Median	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	1000	0	0.00%	561332.42	59578	1024250	561673.00	972277.20	1013585.00	1023415.00	0.98	0.46	0.32
HTTP Request	1000	0	0.00%	561332.42	59578	1024250	561673.00	972277.20	1013585.00	1023415.00	0.98	0.46	0.32

Figure 22:Experiment 4 Iteration 2 – System Performance



APDEX (Application Performance Index)

Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.000	500 ms	1 sec 500 ms	Total
0.000	500 ms	1 sec 500 ms	HTTP Request

Requests Summary

Statistics

Requests	Executions			Response Times (ms)							Throughput	Network (KB/sec)	
Label	#Samples	FAIL	Error %	Average	Min	Max	Median	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	2500	0	0.00%	1390488.86	60183	2691847	1392610.50	2475039.90	2608931.00	2689548.49	0.93	0.44	0.30
HTTP Request	2500	0	0.00%	1390488.86	60183	2691847	1392610.50	2475039.90	2608931.00	2689548.49	0.93	0.44	0.30

Figure 23: Experiment 4 Iteration 3 – System Performance

The outcome of this experiment confirms the stability of the system is as expected. Although system achieves a failure rate of 0, the load of 2500 and even 1000 is can be considered as a heavy load with the available resources according to the response times.

4.5 Conclusion

In each of the experiments, the prototype was able to meet the expectations relevant to accuracy, verifiability, double spent resistance and stability.

CHAPTER 5

CONCLUSION AND FUTURE WORK

This chapter aims to describes the summary of the work done and the findings as well as limitations with the future work that could be done as an extension for this study.

5.1 Summary of the work

This study focuses on achieving transparency and auditability with post voter verification technique. This study proposes a secure method to store and verify votes and the method is based on public key cryptography to achieve anonymity. It uses encrypted private key with voter provided passphrase at the vote submission to encrypt the vote submitted. This technique enables the voter to verify the recorded vote and the submitted vote while protecting the voter anonymity and privacy. The proposed system uses distributed architecture and the system prototype implementation is done using the integration Corda, which is a permissioned blockchain distributed ledger framework. The core language is used to development is Java. System prototype is configured to use two peer nodes and one notary consensus node with REST API to interact with the ledger network and the UI application for user interactions.

The evaluation election scope is set as presidential election and the data set used for the simulation is generated by a java code which randomizes the vote data for each simulation. The evaluation is done for the accuracy, verifiability, double vote casting resistance and stability with the load.

5.2 Findings and Limitations

The prototype system components and simulation tools are deployed in a single computer (PC) which has a processor of Intel(R) Core (TM) i5-8300H CPU @ 2.30GHz and RAM of 15G. The evaluation outcome confirms the prototype is able to meet the functional and non-functional expectations including the accuracy, verifiability, double vote casting resistance and stability with the load. The results show the applicability of the Corda framework as an underlying distributed ledger framework for electronic voting systems.

When it comes to the limitations, prototype is deployed and verified in a personal computer, the hardware resources limit the evaluation parameters such as concurrent thread count, deployable node count etc. With the Corda, the ledger data is not directly visible because of the usage of the corda vault technology which could be identified as a limitation since if there

is a need to query the data, it should use the corda service hub to get the data. There is a possibility of integrating a database such as Postgres but it is not a recommended approach.

5.3 Future work

The proposed system requires to have a voting center to cast the vote but the system could be extended for an uncontrolled environment such as personal computers, smart phones etc. Another aspect is to focus on the authentication mechanisms to extend the system as a full functional system. Also, the evaluation aspects can be enhanced by deploying the system in a production similar environment. Another area can be researching the cryptographic algorithms which could enhance the security of the encrypted voter data. This study and the findings can be used as the basis for developing and end to end electronic voting system.

APPENDICES

Appendix A: Simulation data set samples

```
voting_center,poolingDivision,electoralDistrict,province
ElectoralDistrict01_PD_01_VC_01,ElectoralDistrict01_PD_01,ElectoralDistrict01,Prov
ince01
ElectoralDistrict01_PD_01_VC_02,ElectoralDistrict01_PD_01,ElectoralDistrict01,Prov
ince01
ElectoralDistrict01_PD_02_VC_01,ElectoralDistrict01_PD_02,ElectoralDistrict01,Prov
ince01
ElectoralDistrict01_PD_02_VC_02,ElectoralDistrict01_PD_02,ElectoralDistrict01,Prov
ince01
ElectoralDistrict02_PD_01_VC_01,ElectoralDistrict02_PD_01,ElectoralDistrict02,Prov
ince01
ElectoralDistrict02_PD_01_VC_02,ElectoralDistrict02_PD_01,ElectoralDistrict02,Prov
ince01
ElectoralDistrict02_PD_02_VC_01,ElectoralDistrict02_PD_02,ElectoralDistrict02,Prov
ince01
ElectoralDistrict02_PD_02_VC_02,ElectoralDistrict02_PD_02,ElectoralDistrict02,Prov
ince01
ElectoralDistrict03_PD_01_VC_01,ElectoralDistrict03_PD_01,ElectoralDistrict03,Prov
ince02
ElectoralDistrict03_PD_01_VC_02,ElectoralDistrict03_PD_01,ElectoralDistrict03,Prov
ince02
ElectoralDistrict03_PD_02_VC_01,ElectoralDistrict03_PD_02,ElectoralDistrict03,Prov
ince02
ElectoralDistrict03_PD_02_VC_02,ElectoralDistrict03_PD_02,ElectoralDistrict03,Prov
ince02
ElectoralDistrict04_PD_01_VC_01,ElectoralDistrict04_PD_01,ElectoralDistrict04,Prov
ince02
ElectoralDistrict04_PD_01_VC_02,ElectoralDistrict04_PD_01,ElectoralDistrict04,Prov
ince02
ElectoralDistrict04_PD_02_VC_01,ElectoralDistrict04_PD_02,ElectoralDistrict04,Prov
ince02
ElectoralDistrict04_PD_02_VC_02,ElectoralDistrict04_PD_02,ElectoralDistrict04,Prov
ince02
```

Figure 24: Appendix A.1- Simulation voting center configurations

"{"voter":"0000000008V","transactionId":"06D2FC11C000419D2D43917F9E4BB03637877F0D3BE53936C466957D27F42702","ballotReference":"f6934220-972b-457d-a653-7a90944272e9"}"

"{"voter":"0000000000V","transactionId":"37ACD7A27DDE7658D83D4EA62DAD10CF24AC1E51C1A4ECF77622519C7A350B7D","ballotReference":"ce6fa890-22d6-45b6-85a7-45acfc067549"}"

"{"voter":"0000000014V","transactionId":"F29CA8A710B1FC70391E462138D374B22483C3DD893ED4DBA04E8DDBC8B9EE92","ballotReference":"da2c0005-bfc0-473f-a95e-9adb48ccbd79"}"

"{"voter":"0000000009V","transactionId":"FE8214968C2C96D81784A5E28A1166BDA8CD5999A4266000EE17DDBA62227032","ballotReference":"e3a219d8-8259-488d-842b-bea60d7e2dfa"}"

"{"voter":"0000000004V","transactionId":"526FB97924C2D1C5A360B4880FF6A83C9FB5BED190B12ADCBF39DEA08993E264","ballotReference":"40e25f04-c405-471f-8f95-cfd8dfb7ac29"}"

"{"voter":"0000000011V","transactionId":"A2D56A1C298D957C4A5396DF8C2EF3EB62D850CE6DB0523437496687B8394D8E","ballotReference":"69d9c358-ea9e-43ce-a7ac-7c849bbebec1"}"

"{"voter":"0000000002V","transactionId":"E556EAB894488F987D9443A21BAC233F17456E6CA9C2AEA93A32B0D4EBBC5761","ballotReference":"2461ee50-f8b1-42db-b636-a2ee9a97f22a"}"

"{"voter":"0000000005V","transactionId":"6ADA7087D8162E6EE266611AA786B06546D5E6ABCA95640329A0FEDDE18174FC","ballotReference":"f8491b20-ce32-42c2-879c-89a6ce7a435e"}"

"{"voter":"0000000003V","transactionId":"2B4EC5867CFE7760B1BEBB20652408D1A92E8DD1FF69A839BA6196332483B576","ballotReference":"a7d1512f-126c-4ddf-bbe5-74dab49d7bd6"}"

"{"voter":"0000000012V","transactionId":"3CB5588D8D898AE862FBA04129720EE6146639509F638AAC0B82CBA9EA6CA981","ballotReference":"6796fd70-b83c-470d-a00d-76084ef615e9"}"

"{"voter":"0000000001V","transactionId":"652C1DA97133655CC9F1B20B88CDC4DC78800B0F01C8A1E0217880B9DAD56247","ballotReference":"060299a4-950a-4b38-ba66-dbdd4bd76d09"}"

"{"voter":"0000000013V","transactionId":"CDBFD808E769B3BCDB24A8B306B31861D27B8F47352CE3A1996325484853E579","ballotReference":"7095394e-6bbc-47cb-ba1a-da99a3ce0dc8"}"

"{"voter":"0000000007V","transactionId":"0FF90F6920DC231B495AF3ABE471D9AFE6724DC09E8E5DE382AD60E9C6BF35E6","ballotReference":"2c2f2f02-b178-46a2-9f04-8bfede021d4c"}"

"{"voter":"0000000006V","transactionId":"B45183BC4D7C5E4F34D989675231B64200FBE1E8A7EC484987BA1863A16F5202","ballotReference":"abbc812c-cc03-4c6b-bf1c-c17cb9bedca2"}"

"{"voter":"0000000010V","transactionId":"FDF4C64B510B159E2F1B3E72B9F78CD2D7BCBB137FB3F96D5B91FE1BB36D7FDE","ballotReference":"5cb6e215-9d19-4f2f-8bc6-8b080d7a7353"}"

"{"voter":"0000000016V","transactionId":"EC7FB2FEE948874C37DCEAE5E492E889CBA76772E881963B09A1091F86667E38","ballotReference":"57558bc4-0db6-4e02-86a4-27516387049c"}"

"{"voter":"0000000015V","transactionId":"600DFFDBD29CC3F8760975C89A92C51D5A56BFB08AA7F7680F4C43BA38754DCC","ballotReference":"524ed5a9-c7f6-41df-99ab-2d0b3b7febe7"}"

"{"voter":"0000000017V","transactionId":"92CCB28B238815C26ED30B30598A0187A2567AEEC7F70ED232C93DD335828CFC","ballotReference":"7165a9fa-3624-4aff-aa2b-6ed2f10366c7"}"

"{"voter":"0000000018V","transactionId":"6EA7036AAFA0B6658A5AFB657E54CCB702B7D0E6FD40530A76CF6E71B28D184A","ballotReference":"de0c6993-24e1-44a1-9f20-f5bf623a1116"}"

"{"voter":"0000000019V","transactionId":"E7B4C5D554478071510E393D3541EB0CC45F324ED3C77710A0EFF3535DAEC788","ballotReference":"6ca5fc7d-b4e5-43d1-8881-2e4c92894bbd"}"

"{"voter":"0000000020V","transactionId":"743DBD6E82DA08911E645F15615668570E65A603FD67E956911165444F923298","ballotReference":"fdbc03ee-67e5-48d7-ac0e-5453ad224d3f"}"

"{"voter":"0000000021V","transactionId":"C72C1144A81647E23D7ACC425B13C8077D8D59D32CB4E9CA6348A5574AF0C3AA","ballotReference":"0b7ab400-c0cd-420b-a3a2-edcb95fc58ff"}"

"{"voter":"0000000022V","transactionId":"4626C09DB6D7255B3444DADD91F00FA8A4FAC0CE6F76EE88BB478C2BFA1C6126","ballotReference":"9039f5d2-818c-4e27-9969-5915c6bdd5c7"}"

Figure 25: Appendix A.2 - Vote casting responses - Experiment 1

"{"ballotRef":"0796ec78-d030-4f17-93ab-39fdf521256e","submittedVote":"{\"voterNIC\":\"0000000013V\",\"votingCenterCode\":\"ElectoralDistrict01_PD_01_VC_01\",\"selectedCandidateNic\":\"100000000V\",\"votedTimestamp\":\"2024-02-23T15:02:28.771\"}","recordedVote":"{\"candidateId\":\"cccac7fc-5be9-44f7-8bb5-0337a96edd49\",\"candidateDetails\":\"{\\\"id\\\":\\\"cccac7fc-5be9-44f7-8bb5-0337a96edd49\\\",\\\"name\\\":\\\"Candidate_01\\\",\\\"nic\\\":\\\"100000000V\\\",\\\"politicalParty\\\":\\\"Party 01\\\"}\"}","verification":"VERIFIED"}"

"{"ballotRef":"578d0c8e-5e3b-4adb-b976-ea49148a79b2","submittedVote":"{\"voterNIC\":\"0000000009V\",\"votingCenterCode\":\"ElectoralDistrict01_PD_02_VC_01\",\"selectedCandidateNic\":\"300000000V\",\"votedTimestamp\":\"2024-02-23T15:02:28.737\"}","recordedVote":"{\"candidateId\":\"234e7927-ecf1-4ad0-912f-1731226733a8\",\"candidateDetails\":\"{\\\"id\\\":\\\"234e7927-ecf1-4ad0-912f-1731226733a8\\\",\\\"name\\\":\\\"Candidate_03\\\",\\\"nic\\\":\\\"300000000V\\\",\\\"politicalParty\\\":\\\"Party 03\\\"}\"}","verification":"VERIFIED"}"

"{"ballotRef":"9986a691-917c-447f-ba0b-6f4a4cb50c0a","submittedVote":"{\"voterNIC\":\"0000000012V\",\"votingCenterCode\":\"ElectoralDistrict01_PD_01_VC_01\",\"selectedCandidateNic\":\"100000000V\",\"votedTimesta

mp\":\"2024-02-23T15:02:28.758\"}","recordedVote":"{\"candidateId\":\"cccac7fc-5be9-44f7-8bb5-0337a96edd49\",\"candidateDetails\":\"{\\\"id\\\":\\\"cccac7fc-5be9-44f7-8bb5-0337a96edd49\\\",\\\"name\\\":\\\"Candidate_01\\\",\\\"nic\\\":\\\"100000000V\\\",\\\"politicalParty\\\":\\\"Party 01\\\"}\"}","verification":"VERIFIED"}"

"{"ballotRef":"ad4684de-002d-4ed7-adc4-a88e74036d0a","submittedVote":"{\"voterNIC\":\"0000000008V\",\"votingCenterCode\":\"ElectoralDistrict04_PD_02_VC_01\",\"selectedCandidateNic\":\"200000000V\",\"votedTimestamp\":\"2024-02-23T15:02:28.737\"}","recordedVote":"{\"candidateId\":\"bd9ce9a4-b662-429c-8d36-75ea0bf3c895\",\"candidateDetails\":\"{\\\"id\\\":\\\"bd9ce9a4-b662-429c-8d36-75ea0bf3c895\\\",\\\"name\\\":\\\"Candidate_02\\\",\\\"nic\\\":\\\"200000000V\\\",\\\"politicalParty\\\":\\\"Party 02\\\"}\"}","verification":"VERIFIED"}"

"{"ballotRef":"c03e2865-d473-409a-9536-baf36c052d48","submittedVote":"{\"voterNIC\":\"0000000004V\",\"votingCenterCode\":\"ElectoralDistrict02_PD_01_VC_02\",\"selectedCandidateNic\":\"200000000V\",\"votedTimestamp\":\"2024-02-23T15:02:28.737\"}","recordedVote":"{\"candidateId\":\"bd9ce9a4-b662-429c-8d36-75ea0bf3c895\",\"candidateDetails\":\"{\\\"id\\\":\\\"bd9ce9a4-b662-429c-8d36-75ea0bf3c895\\\",\\\"name\\\":\\\"Candidate_02\\\",\\\"nic\\\":\\\"200000000V\\\",\\\"politicalParty\\\":\\\"Party 02\\\"}\"}","verification":"VERIFIED"}"

"{"ballotRef":"8b482a8f-fdeb-4dc3-a5a8-72e63aa5988f","submittedVote":"{\"voterNIC\":\"0000000003V\",\"votingCenterCode\":\"ElectoralDistrict03_PD_01_VC_01\",\"selectedCandidateNic\":\"300000000V\",\"votedTimestamp\":\"2024-02-23T15:02:28.737\"}","recordedVote":"{\"candidateId\":\"234e7927-ecf1-4ad0-912f-1731226733a8\",\"candidateDetails\":\"{\\\"id\\\":\\\"234e7927-ecf1-4ad0-912f-1731226733a8\\\",\\\"name\\\":\\\"Candidate_03\\\",\\\"nic\\\":\\\"300000000V\\\",\\\"politicalParty\\\":\\\"Party 03\\\"}\"}","verification":"VERIFIED"}"

"{"ballotRef":"161f38c4-e8ad-40de-ad30-dca0534b7bfd","submittedVote":"{\"voterNIC\":\"0000000011V\",\"votingCenterCode\":\"ElectoralDistrict01_PD_02_VC_01\",\"selectedCandidateNic\":\"200000000V\",\"votedTimestamp\":\"2024-02-23T15:02:28.754\"}","recordedVote":"{\"candidateId\":\"bd9ce9a4-b662-429c-8d36-75ea0bf3c895\",\"candidateDetails\":\"{\\\"id\\\":\\\"bd9ce9a4-b662-429c-8d36-75ea0bf3c895\\\",\\\"name\\\":\\\"Candidate_02\\\",\\\"nic\\\":\\\"200000000V\\\",\\\"politicalParty\\\":\\\"Party 02\\\"}\"}","verification":"VERIFIED"}"

"{"ballotRef":"43d1b27c-aca5-4752-8175-b48db6723f92","submittedVote":"{\"voterNIC\":\"0000000006V\",\"votingCenterCode\":\"ElectoralDistrict02_PD_01_VC_01\",\"selectedCandidateNic\":\"200000000V\",\"votedTimestamp\":\"2024-02-23T15:02:28.737\"}","recordedVote":"{\"candidateId\":\"bd9ce9a4-b662-429c-8d36-75ea0bf3c895\",\"candidateDetails\":\"{\\\"id\\\":\\\"bd9ce9a4-b662-429c-8d36-75ea0bf3c895\\\",\\\"name\\\":\\\"Candidate_02\\\",\\\"nic\\\":\\\"200000000V\\\",\\\"politicalParty\\\":\\\"Party 02\\\"}\"}","verification":"VERIFIED"}"

Figure 26: Appendix A.3 - Verification responses - Experiment 2

Appendix B: Source codes

```
process("Register Election", () -> {
        ElectionModel model = new ElectionModel();
        model.setElectionType("Presidential Election");
        model.setElectionDate(LocalDate.now());
        model.setStartTime(LocalTime.now());
        model.setEndTime(LocalTime.now().plusHours(8));
        model.setStatus(ElectionStatus.ACTIVE.name());
        adminService.createElection(model);
    });

    ElectionModel election = adminService.getElections().get(0);

    process("Create Political Parties", () -> {
        PoliticalPartyModel p1 = new PoliticalPartyModel();
        p1.setName("Party 01");
        p1.setElection(election);
        adminService.createPoliticalParty(p1);

        PoliticalPartyModel p2 = new PoliticalPartyModel();
        p2.setName("Party 02");
        p2.setElection(election);
        adminService.createPoliticalParty(p2);

        PoliticalPartyModel p3 = new PoliticalPartyModel();
        p3.setName("Party 03");
        p3.setElection(election);
        adminService.createPoliticalParty(p3);
    });


    process("Create Provinces", () -> {
        List<String> provinces =
Arrays.stream(Province.values()).map(Enum::name).collect(Collectors.toList());
        for (String name : provinces) {
            ProvinceModel p1 = new ProvinceModel();
            p1.setName(name);
            adminService.createProvince(p1);
        }
    });

    process("Create Electoral Districts", () -> {
        List<ProvinceModel> provinces = adminService.getProvinces();
        for (ProvinceModel province : provinces) {
            List<ElectoralDistrict> electoralDistricts =
Province.valueOf(province.getName()).electoralDistricts;
            for (ElectoralDistrict ed : electoralDistricts) {
```

```
            ElectoralDistrictModel model = new ElectoralDistrictModel();
            model.setName(ed.name());
            model.setProvince(province);
            adminService.createElectoralDistrict(model);

        }
    }
});

process("Create Pooling Divisions", () -> {
    List<ElectoralDistrictModel> electoralDistricts = adminService.getElectoralDistricts();
    for (ElectoralDistrictModel districtModel : electoralDistricts) {
        PoolingDivisionModel pd1 = new PoolingDivisionModel();
        pd1.setName(districtModel.getName() + "_PD_01");
        pd1.setElectoralDistrict(districtModel);
        adminService.createPoolingDivision(pd1);

        PoolingDivisionModel pd2 = new PoolingDivisionModel();
        pd2.setName(districtModel.getName() + "_PD_02");
        pd2.setElectoralDistrict(districtModel);
        adminService.createPoolingDivision(pd2);
    }
});

process("Create Voting Centers", () -> {
    List<PoolingDivisionModel> poolingDivisions = adminService.getPoolingDivisions();
    for (PoolingDivisionModel model : poolingDivisions) {
        VotingCenterModel vc1 = new VotingCenterModel();
        vc1.setCode(model.getName() + "_VC_01");
        vc1.setElection(election);
        vc1.setPoolingDivision(model);
        adminService.createVotingCenter(vc1);

        VotingCenterModel vc2 = new VotingCenterModel();
        vc2.setCode(model.getName() + "_VC_02");
        vc2.setElection(election);
        vc2.setPoolingDivision(model);
        adminService.createVotingCenter(vc2);
    }
});

writeMetaDataToFile();

process("Create candidates", () -> {
    List<PoliticalPartyModel> politicalParties = adminService.getPoliticalParties();
    CandidateModel c1 = new CandidateModel();
    c1.setName("Candidate_01");
    c1.setNic("100000000V");
    c1.setPoliticalParty(politicalParties.get(0));
```

```
        adminService.createCandidate(c1);

        CandidateModel c2 = new CandidateModel();
        c2.setName("Candidate_02");
        c2.setNic("200000000V");
        c2.setPoliticalParty(politicalParties.get(1));
        adminService.createCandidate(c2);

        CandidateModel c3 = new CandidateModel();
        c3.setName("Candidate_03");
        c3.setNic("300000000V");
        c3.setPoliticalParty(politicalParties.get(2));
        adminService.createCandidate(c3);
      });

      process("Create Voters", () -> {
        List<VotingCenterModel> votingCenters = adminService.getVotingCenters();

        for (int i = 0; i < 100; i++) {
          int nextInt = random.nextInt(votingCenters.size());
          VoterModel v1 = getVoter(i, votingCenters.get(nextInt));
          adminService.createVoter(v1);
        }

      });
```

Figure 27: Appendix B.1 – Simulation configuration generation source code

```
public class BallotCastFlow {

  private BallotCastFlow() {
  }

  @InitiatingFlow
  @StartableByRPC
  public static class Initiator extends BaseCreateFlowInitiator<BallotDTO, BallotState> {

    private static final Logger logger =
LoggerFactory.getLogger(BallotCastFlow.Initiator.class);

    private final SubmittedVoteDTO submittedVoteDTO;
    private final String privateKeyPassphrase;

    public Initiator(SubmittedVoteDTO submittedVoteDTO, String privateKeyPassphrase) {
      super(new BallotDTO());
      this.submittedVoteDTO = submittedVoteDTO;
      this.privateKeyPassphrase = privateKeyPassphrase;
    }
```

```
    @Override
    public BallotState getState(Party currentNode, Set<Party> allPeers) {
      return new BallotState(getDto(), currentNode, allPeers);
    }

    @Suspendable
    @Override
    public void executePreSubFlows(FlowResult result) throws FlowException {
      try {

        fillBallotDTO();

        CandidateDTO candidateByNic =
DataFlowService.getCandidateByNic(getServiceHub(),
submittedVoteDTO.getSelectedCandidateNic());

        CandidateVoteDTO candidateVoteDTO = new CandidateVoteDTO();
        candidateVoteDTO.setCandidateId(candidateByNic.getIdentifier());

        CandidateVoteRecordFlow.Initiator recordFlow = new
CandidateVoteRecordFlow.Initiator(candidateVoteDTO);
        SignedTransaction transaction = subFlow(recordFlow).getTransaction();
        BaseState<CandidateVoteDTO> candidateVoteState =
(BaseState<CandidateVoteDTO>) transaction.getTx().getOutput(0);

        String countedVoteReference = candidateVoteState.getDto().getIdentifier();


        KeyPair keyPair = CryptoHelper.generateKeyPair();
        byte[] privateKey = CryptoHelper.encryptPrivateKey(keyPair.getPrivate(),
this.privateKeyPassphrase);

        getDto().setPrivateKey(privateKey);

        byte[] encrypted =
CryptoHelper.encrypt(getPayloadForEncryption(countedVoteReference),
keyPair.getPublic());
        Vote vote = new Vote();
        vote.setValue(encrypted);
        getDto().setVote(vote);

getDto().setVotedTimestamp(TimestampHelper.getLocalDataTimeToString(LocalDateTime.n
ow()));

      } catch (Exception e) {
        throw new FlowException(e);
      }
    }
```

```
    private void fillBallotDTO() throws FlowException {

getDto().setElectionId(DataFlowService.getActiveElection(getServiceHub()).getIdentifier());
        getDto().setVoterId(DataFlowService.getVoterByNic(getServiceHub(),
submittedVoteDTO.getVoterNIC()).getIdentifier());

getDto().setVotingCenterId(DataFlowService.getVotingCenterByCode(getServiceHub(),
submittedVoteDTO.getVotingCenterCode()).getIdentifier());
    }

    @NotNull
    private String getPayloadForEncryption(String countedVoteReference) {
      JsonObject jsonObject = Json.createObjectBuilder()
            .add("submittedVote", getSubmittedVoteAsJson())
            .add("recordedVoteRef", countedVoteReference)
            .build();
      return jsonObject.toString();
    }

    private String getSubmittedVoteAsJson() {
      JsonObject jsonObject = Json.createObjectBuilder()
            .add("voterNIC", submittedVoteDTO.getVoterNIC())
            .add("votingCenterCode", submittedVoteDTO.getVotingCenterCode())
            .add("selectedCandidateNic", submittedVoteDTO.getSelectedCandidateNic())
            .add("votedTimestamp", submittedVoteDTO.getTimestamp().toString())
            .build();
      return jsonObject.toString();

    }

    public Command getCommand() {
      return new BallotContract.Cast();
    }

    @Override
    public List<Rule> validationRules() throws FlowException {
      try {
        Rule isWithinTheElectionTimeFrame = () -> {
          ElectionDTO dto = DataFlowService.getActiveElection(getServiceHub());
          if (!isValidCastingTime(dto)) {
            throw new FlowException("Invalid ballot casting time frame");
          }
        };

        Rule eligibleToCast = () -> {
          VoterDTO voter = DataFlowService.getVoterByNic(getServiceHub(),
submittedVoteDTO.getVoterNIC());

          if (!VoterStatus.ACTIVE.equals(voter.getStatus())) {
```

```
                throw new FlowException("Not a eligible voter");
            }

            Optional<BallotDTO> ballotCastedByVoter =
DataFlowService.getBallotCastedByVoter(getServiceHub(), voter.getIdentifier());

            if (ballotCastedByVoter.isPresent()) {
                throw new FlowException("Vote already casted by the voter.");
            }
          };

          return Arrays.asList(isWithinTheElectionTimeFrame, eligibleToCast);
        } catch (Exception e) {
          throw new FlowException(e);
        }
      }

      private boolean isValidCastingTime(ElectionDTO dto) {
        LocalDate electionDate = LocalDate.parse(dto.getDate());
        Instant currentTime = getServiceHub().getClock().instant();

        LocalDateTime currentDateTime = LocalDateTime.ofInstant(currentTime,
ZoneId.systemDefault());
        LocalDateTime electionStartDateTime = LocalDateTime.of(electionDate,
LocalTime.parse(dto.getStartTime()));
        LocalDateTime electionEndDateTime = LocalDateTime.of(electionDate,
LocalTime.parse(dto.getEndTime()));

        return currentDateTime.isAfter(electionStartDateTime) &&
currentDateTime.isBefore(electionEndDateTime);
      }

      @Override
      public String getContractId() {
        return BallotContract.ID;
      }
    }

    @InitiatedBy(BallotCastFlow.Initiator.class)
    public static class Responder extends BaseCreateFlowResponder {
      public Responder(FlowSession otherPartySession) {
        super(otherPartySession);
      }
    }
}
```

Figure 28: Appendix B.2 - Ballot cast flow source code

```
public class BallotContract implements Contract {
```

```
    public static final String ID = "org.chainballotx.contracts.BallotContract";

    @Override
    public void verify(@NotNull LedgerTransaction tx) throws IllegalArgumentException {

        CommandWithParties<CommandData> command =
requireSingleCommand(tx.getCommands(), CommandData.class);

        requireThat(require -> {
            require.using("Not a valid command.", verifyCommand(command));
            return null;
        });
    }

    private boolean verifyCommand(CommandWithParties<CommandData> command) {
        return (command.getValue() instanceof Cast);
    }

    public static class Cast implements Command {
    }
}
```

Figure 29: Appendix B.3 - Ballot contract source code

```
@CordaSerializable
public class BallotDTO extends BaseDTO implements Serializable {
    private String voterId;
    private String electionId;
    private String votingCenterId;

    // flow initialises the below
    private Vote vote;
    private String votedTimestamp;

    private byte[] privateKey;

    public String getVoterId() {
        return voterId;
    }

    public void setVoterId(String voterId) {
        this.voterId = voterId;
    }

    public String getElectionId() {
        return electionId;
    }
```

```
    public void setElectionId(String electionId) {
        this.electionId = electionId;
    }

    public String getVotingCenterId() {
        return votingCenterId;
    }

    public void setVotingCenterId(String votingCenterId) {
        this.votingCenterId = votingCenterId;
    }

    public Vote getVote() {
        return vote;
    }

    public void setVote(Vote vote) {
        this.vote = vote;
    }

    public String getVotedTimestamp() {
        return votedTimestamp;
    }

    public void setVotedTimestamp(String votedTimestamp) {
        this.votedTimestamp = votedTimestamp;
    }

    public void setPrivateKey(byte[] privateKey) {
        this.privateKey = privateKey;
    }

    public byte[] getPrivateKey() {
        return privateKey;
    }
}
```

Figure 30: Appendix B.4 - Ballot DTO source code

```
@BelongsToContract(BallotContract.class)
public class BallotState extends BaseState<BallotDTO> {


    public BallotState(BallotDTO dto, Party currentNode, Set<Party> otherParticipants) {
        super(dto, currentNode, otherParticipants);
    }

    @NotNull
```

```
    @Override
    public PersistentState generateMappedObject(@NotNull MappedSchema schema) {
        if (schema instanceof ChainBallotXSchemaV1) {
            return new BallotStateEntity(this.getDto(), this.getLinearId());
        }
        throw new IllegalArgumentException("Unsupported Schema");
    }
}
```

Figure 31: Appendix B.5 - Ballot state source code

REFERENCES

Nakamoto, S., 2008. Bitcoin: A Peer-to-Peer Electronic Cash System.

Anwar ul Hassan, C., Hammad, M., Iqbal, J., Hussain, S., Ullah, S.S., AlSalman, H., Mosleh, M.A.A., Arif, M., 2022. A Liquid Democracy Enabled Blockchain-Based Electronic Voting System. Scientific Programming 2022, 1–10. https://doi.org/10.1155/2022/1383007.

Yacoubi, A., Erraha, B., Asri, H., 2021. An Electronic Voting System adopting Blockchain: Interpretation, Characteristics and Investigation. E3S Web Conf. 297, 01076. https://doi.org/10.1051/e3sconf/202129701076.

Xiao, S., Wang, X.A., Wang, W., Wang, H., 2020. Survey on Blockchain-Based Electronic Voting, in: Barolli, L., Nishino, H., Miwa, H. (Eds.), Advances in Intelligent Networking and Collaborative Systems, Advances in Intelligent Systems and Computing. Springer International Publishing, Cham, pp. 559–567. https://doi.org/10.1007/978-3-030-29035-1_54.

Jafar, U., Aziz, M.J.A., Shukur, Z., 2021. Blockchain for Electronic Voting System—Review and Open Research Challenges. Sensors 21, 5874. https://doi.org/10.3390/s21175874.

Majumder, S., Ray, S., Sadhukhan, D., Dasgupta, M., Das, A.K., Park, Y., 2024. ECC-EXONUM-eVOTING: A Novel Signature-Based e-Voting Scheme Using Blockchain and Zero Knowledge Property. IEEE Open J. Commun. Soc. 5, 583–598. https://doi.org/10.1109/OJCOMS.2023.3348468.

Chowdhury, M.J.M., Ferdous, Md.S., Biswas, K., Chowdhury, N., Kayes, A.S.M., Alazab, M., Watters, P., 2019. A Comparative Analysis of Distributed Ledger Technology Platforms. IEEE Access 7, 167930–167943. https://doi.org/10.1109/ACCESS.2019.2953729.

Mohammad Hosam Sedky, Ramzy Hamed, E.M., 2015. A secure e-Government's e-voting system, in: 2015 Science and Information Conference (SAI). Presented at the 2015 Science and Information Conference (SAI), IEEE, London, pp. 1365–1373. https://doi.org/10.1109/SAI.2015.7237320.

IFES Election Guide | Country Profile: Sri Lanka [WWW Document], n.d. URL https://www.electionguide.org/countries/id/201/ (accessed 11.11.23).

ICTs in Elections Database | International IDEA [WWW Document], n.d. URL https://www.idea.int/data-tools/data/icts-elections-database (accessed 11.12.23).

Jafar, U., Ab Aziz, M.J., Shukur, Z., Hussain, H.A., 2022. A Systematic Literature Review and Meta-Analysis on Scalable Blockchain-Based Electronic Voting Systems. Sensors 22, 7585. https://doi.org/10.3390/s22197585.

Permissioned and Permissionless Blockchains: A Comprehensive Guide, 2019. URL https://www.blockchain-council.org/blockchain/permissioned-and-permissionless-blockchains-a-comprehensive-guide/ (accessed 11.12.23).

Electronic Voting - Case Study: Brazil [WWW Document], n.d. URL https://cs.stanford.edu/people/eroberts/cs201/projects/2006-07/electronic-voting/index_files/page0006.html (accessed 2.10.24).

Electronic voting in the United States, 2024. . Wikipedia.

Pihlak, H., 2019. i-Voting – the Future of Elections? [WWW Document]. e-Estonia. URL https://e-estonia.com/i-voting-the-future-of-elections/ (accessed 2.11.24).

Underlying principles of the i-voting system in Estonia - Smartmatic [WWW Document], n.d. . Smartmatic.com. URL http://www.smartmatic.com/case-studies/article/underlying-principles-of-the-i-voting-system-in-estonia/ (accessed 2.11.24).

Taş, R., Tanrıöver, Ö.Ö., 2021. A Manipulation Prevention Model for Blockchain-Based E-Voting Systems. Security and Communication Networks 2021, 1–16. https://doi.org/10.1155/2021/6673691.

Küsters, R., Liedtke, J., Müller, J., Rausch, D., Vogt, A., 2020. Ordinos: A Verifiable Tally-Hiding E-Voting System, in: 2020 IEEE European Symposium on Security and Privacy (EuroS&P). Presented at the 2020 IEEE European Symposium on Security and Privacy (EuroS&P), IEEE, Genoa, Italy, pp. 216–235. https://doi.org/10.1109/EuroSP48549.2020.00022.

Corda Community Edition Key Concepts - Community & Open Source 4.11 [WWW Document], 2024. . R3 Documentation. URL https://docs.r3.com/en/platform/corda/4.11/community/about-corda/corda-key-concepts.html (accessed 2.17.24).

Ministry of Defence - Sri Lanka [WWW Document], n.d. URL https://www.defence.lk/Article/view_article/27044 (accessed 2.12.24).