VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgaum-590018

A PROJECT REPORT **(15CSP85)** ON

“Secure Online Voting System Using Hyperledger BlockChain Fabric”

Submitted in Partial fulfillment of the Requirements for the Degree of

Bachelor of Engineering in Computer Science & Engineering

By

DYLAN DORNIN PINHEIRO (1CR16CS046)

MONISHA RAJESH (1CR16CS094)

PRAJWAL PANDIT KHOT(1CR16CS117)

R SHIV NARAYAN REDDY(1CR16CS122)

Under the Guidance of,

Internal Guide	**External Guide**
Dr. Rijo Jackson Tom	**Mr. Nagesha S P**
Assistant Professor	**Scientist ‘F’**
Dept. of CSE	**LRDE, DRDO**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled **"Secure Online Voting System using Hyperledger Blockchain Fabric"** carried out by **Mr. Dylan Dornin Pinheiro**, USN **1CR16CS046**, **Ms. Monisha Rajesh**, USN **1CR16CS094**, **Mr. Prajwal Pandit Khot**, USN **1CR16CS117, Mr. R Shiv Narayan Reddy**, USN **1CR16CS122,** bonafide students of CMR Institute of Technology, in partial fulfillment for the award of **Bachelor of Engineering** in Computer Science and Engineering of the Visveswaraiah Technological University, Belgaum during the year 2019-2020. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

.

________________	________________	________________
Dr. Rijo Jackson Tom	**Dr. Prem Kumar Ramesh**	**Dr. Sanjay Jain**
Assistant Professor	**Professor & Head**	**Principal**
Dept. of CSE, CMRIT	**Dept. of CSE, CMRIT**	**CMRIT**

External Viva

Name of the examiners	Signature with date
1.	________________
2.	________________

DECLARATION

We, the students of Computer Science and Engineering, CMR Institute of Technology, Bangalore declare that the work entitled "**Secure Online Voting System using Hyperledger Blockchain Fabric** " has been successfully completed under the guidance of Prof. Rijo Jackson Tom, Computer Science and Engineering Department, CMR Institute of technology, Bangalore. This dissertation work is submitted in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science and Engineering during the academic year 2019 - 2020. Further the matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place:

Date:

Team members:

Dylan Dornin Pinheiro (1CR16CS046) ____________________

Monisha Rajesh (1CR16CS094) ____________________

Prajwal Pandit Khot (1CR16CS117) ____________________

R Shiv Narayan Reddy (1CR16CS122) ____________________

ABSTRACT

Secure Voting is a web-based online voting system that will help you organize elections in a quick, convenient, and cost-effective way. This also improves voter turnout and build instant results, using this system a person can also vote from outside of his/her allotted constituency and the tallying of the votes will be done automatically, thus saving time and enabling the Election Commissioner of India to announce the result within a very short period. We have implemented this system using Hyperledger Fabric 2.0 for the back-end and used NodeJS for the front-end. This is a web application that can be run on the systems provided at the polling booths during the elections.

ACKNOWLEDGEMENT

We take this opportunity to express our sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing us a platform to pursue our studies and carry out our final year project

We have a great pleasure in expressing our deep sense of gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for his constant encouragement.

We would like to thank **Dr. Prem Kumar Ramesh,** Professor and Head, Department of Computer Science and Engineering, CMRIT, Bangalore, who has been a constant support and encouragement throughout the course of this project.

We consider it a privilege and honor to express our sincere gratitude to our guide **Dr. Rijo Jackson Tom,** Assistant Professor**,** Department of Computer Science and Engineering, for the valuable guidance throughout the tenure of this review.

We heartily thank our external project guide, **Mr. Nagesha S P**, Scientist F, LRDE, DRDO for his guidance and suggestions during this project work.

We also extend our thanks to all the faculty of Computer Science and Engineering who directly or indirectly encouraged me.

Finally, we would like to thank our parents and friends for all their moral support they have given us during the completion of this work.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

India is a constitutional democracy with a parliamentary system of government, and at the heart of the system is a commitment to hold regular, free and fair elections. These elections determine the composition of the government, the membership of the two houses of parliament, the state and union territory legislative assemblies, and the Presidency and vice-presidency.

The aim of the project is to create and manage polling and election details. This is a system which enables all citizens to cast their vote online. Increasing the voting percentage across the country is the major goal of this project. People have to visit their allotted booth to cast their vote in the present system across the country. This system is online and hence even people who live out of their home town can also vote by visiting any of the booths where they stay. Database of all the eligible citizens and candidates are maintained by the system. Online voting system will allow an Indian citizen residing abroad to enrol in voter’s list and exercise his/her vote.

Thus, our project, online voting should enhance the opportunities of voters outside the country to vote for their nation on implementation.

1.1 Relevance of the Project

Modern democracies are built up on voting system, whether traditional ballot based or electronic voting. In recent years, voter apathy (lack of interest) has been increasing, especially among the younger computer/techno savvy generation. Evoting is pushed as a potential solution to attract young voters. For a robust e-voting scheme, a number of functional and security requirements are specified including transparency, accuracy, auditability, system and data integrity, secrecy/privacy, availability, and distribution of authority.

Existing works explore how blockchain can be used to improve the e-voting schemes or provide some strong guarantees of the above listed requirements. However, these papers do not discuss the implementation challenges and limitations of the blockchain (and smart contract) technologies at their current state to fully support a large scale voting scheme.

1.1.1 Drawbacks of current voting system

The problem of just printing up all the ballots ahead of time and getting them right. Then there's the issue of each district will have different ballots. And so on and so forth. And this is apparently a fairly large problem.

There are validity problems - what if the person accidentally checks two boxes on the same race without noticing it (the butterfly ballot in Florida in 2000 is a poster child for this). What if they don't check any box - that may or may not be intentional?

In addition, people who are uneducated and can be manipulated into voting for someone that might not benefit them

The danger for EVM manipulations is not just from its software. Even the hardware isn’t safe. Alex Halderman, professor of computer science in the University of Michigan says, “EVMs used in the West require software attacks as they are sophisticated voting machines and their hardware cannot be replaced cheaply. In contrast, the Indian EVMs can easily be replaced either in part or as wholesale units.”

The EVM manufacturers developed an “Authentication Unit” engaging the services of SecureSpin, a Bangalore based software services firm. The Unit was developed and tested in 2006 but when the project was ready for implementation, the project was mysteriously shelved at the instance of the Election Commission. Several questions posed to the Election Commission for taking this decision went unanswered.

1.1.2 Why do we need E-voting?

To address voter tampering, blockchains generate cryptographically secure voting records. Votes are recorded accurately, permanently, securely, and transparently. So, no one can modify or manipulate votes. Furthermore, blockchains preserve participant’s anonymity while still being open to public inspection. Although nothing is totally secure, tampering is nearly impossible with blockchains.

BEV might promote more voter participation. For instance, corporate annual general meetings can be costly events with low shareholder participation. With increasing cross border investments, companies face pressure to increase in investor engagement. BEV is a flexible solution that enables secure, cost-effective voting to facilitate shareholder participation and voting from a distance.

BEV can increase the speed with which votes are tallied. For example, Agora reported that it published election results on its website five days before the official manual counts ended.

BEV can eliminate ambiguities. For example, in the 2017 Virginia House of Delegates election, the winner was chosen from paper ballots placed in a bowl. One vote initially wasn't counted because that voter made confusing marks on the ballot. Such ambiguity is less likely to arise with BEV.

Finally, with BEV, individual votes will be publicly available, while voters are masked behind an encrypted key. This offers greater privacy and security than traditional ballot boxes and could reduce voter suppression. Bad actors can't identify voters and therefore can't target them.

1.1.3 Previous Work

France

In 2012, authorities in France allowed their citizens who live abroad to vote online through a web portal that has been developed by Scytl SA, a private company founded as a university spin-off. The online voting processes has been later cancelled by National Cybersecurity Agency due to increasing risks of cyber-attacks.

Brazil

In Brazil, voting machines are in use since 1996. These machines, also called kiosks, are modified computers that run specially designed software, which stores and counts the number of votes entered by the attached keypad.

Estonia

Estonia, on the other hand, provides a full e-voting solution for the governmental elections. Every citizen in Estonia possesses an ID card equipped with a digital chip that contains identity and biometric information. People who want to attend online elections, which are held in parallel with conventional elections made in vote centers, should use their ID card and a card reader (provided by the authorities) to login the specially designed web portal or e-voting application, called i-Voting. It is also possible to use a mobile ID, that involves an SMS authentication, in later versions. The i-Voting application can be reached via its web portal, www.valimised.ee, during the elections period. It remains active for several days, until the traditional election's day (which is only 1 day). The ID card acts as a token (possessive security factor) and the voter should also type a unique predefined password (knowledge factor) for authentication himself/herself to the system.

Australia

iVote is a remote electronic voting system in New South Wales that allows eligible voters a chance to vote over the Internet. However, during the New South Wales state election in 2015, there were several reports that over 66,000 electronic votes could have been compromised. Although the iVote website is secure, security specialist believe that a third party website was able to attack the system. This was the first time a major vulnerability was discovered in the middle of an ongoing poll.

1.1.4 Scope of the Project

Governments and other stakeholders will need to address several major challenges before blockchain sees widespread use for e-voting. Although blockchains are good at providing security and accuracy, public confidence and trust are necessary ingredients for BEV's success. Blockchain's complexity might hinder mainstream public acceptability of BEV. Broadband access and digital user skills are also concerning.

In 2016, the non-profit Democracy Earth Foundation used a blockchain to give Colombian expatriates a voice in the 2016 peace plebiscite that was conducted to ratify the agreement to terminate the conflict between the Colombian government and FARC guerrillas. According to the foundation, a main challenge in the deployment blockchain is the technology's immaturity.

Let's now consider software quality. Estimates have suggested that, on average, there are from 15 to 50 defects per 1,000 LOC. For Ethereum, the blockchain-based distributed-computing platform used by Moscow's Active Citizen program (which features smart contracts), the number might be twice that. This might be attributed to Ethereum's immaturity. The Economist quoted a blogger who said that Ethereum contracts are “candy for hackers.”29 Also, sufficient observations haven’t yet been accumulated to determine blockchain-based platforms’ scalability.

Traditional voting emphasizes the authority of the state. BEV emphasizes voter transparency. The BEV process is transparent, decentralized, and bottom-up. BEV might not perform well in a society whose culture and values exhibit low compatibility with these values.

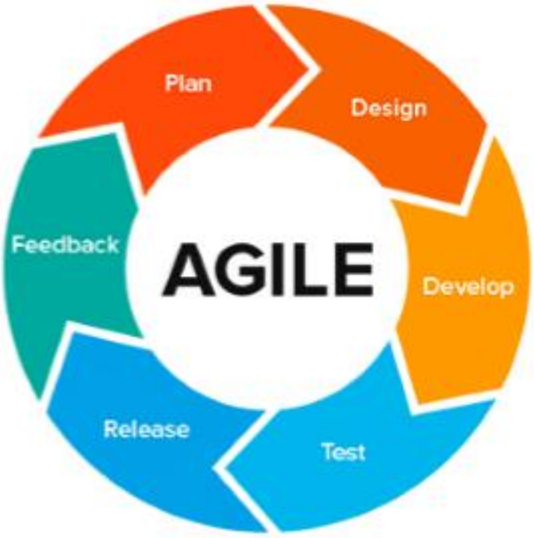
Also, **blockchains** require much energy to perform authentication and validation, and they’re slow. So, using them for national e-voting might not be practical yet.

Finally, BEV will shift power away from central actors such as electoral authorities and government agencies. Thus, the technology is likely to face resistance from political leaders who benefit from the status.

1.2 Methodology

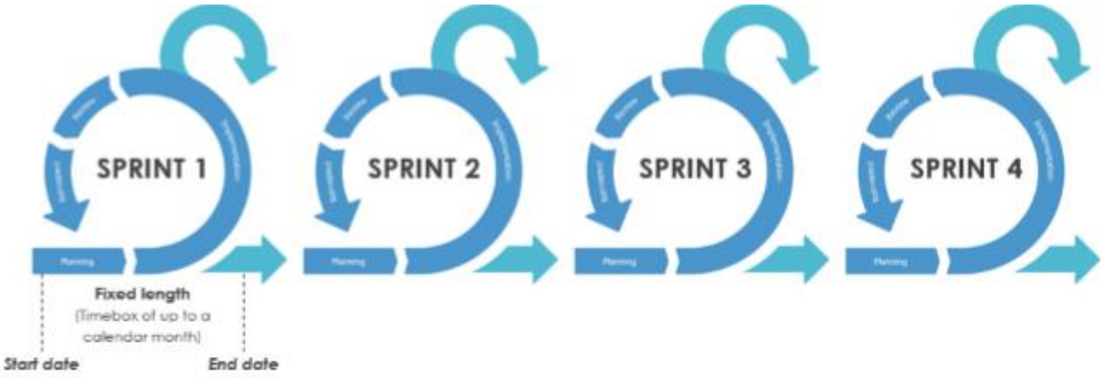
Agile

Agile is a process by which a team can manage a project by breaking it up into several stages and involving constant collaboration with stakeholders and continuous improvement and iteration at every stage. It promotes continuous iteration of development and testing throughout the software development lifecycle of the project. Both development and testing activities are concurrent.



Scrum

SCRUM is an agile development method which concentrates specifically on how to manage tasks within a team-based development environment. Scrum encourages teams to learn through experiences, self-organize while working on a problem



The Main Artefacts

- Product Backlog is the master list of work that needs to get done maintained by the product owner or product manager.
- Sprint Backlog is the list of items, user stories, or bug fixes, selected by the development team for implementation in the current sprint cycle.
- Increment (or Sprint Goal) is the usable end-product from a sprint.

CHAPTER 2

LITERATURE SURVEY

2.1 Decentralized e-Voting System

Harsha V. Patil, Kanchan G. Rathi, Malati V.Tribhuwan, “A Study on Decentralized E-Voting System Using Blockchain Technology”, Dept. of Computer Science, D .Y. Patil ACS College, Volume: 05 Issue: 11 | Nov 2018, e-ISSN: 2395-0056

This paper [1] talks about the blockchain technology may address many issues regarding e-voting schemes and make e-voting cheaper, easier, and much more secure to implement. It is a considerably new paradigm that can help to form decentralized systems, which assure the data integrity, availability, and fault tolerance. Some state that “the blockchain technology is bringing us the Internet of value: a new, distributed platform that can help us reshape the world of business and transform the old order of human affairs for the better.”. The blockchain systems are formed as decentralized networked systems of computers, which are used for validating and recording the pure online transactions.

- The first transaction added to the block will be a special transaction that represents the candidate.
- When this transaction is created it will include the candidate's name and will serve as the foundation block, with every vote for that specific candidate placed on top of it. Unlike the other transactions, the foundation will not count as a vote, and it will only contain the name of the candidate.
- Every time a person votes the transaction gets will be recorded and the blockchain will be updated. To ensure that the system is secure, the block will contain the previous voter’s information.

If any of the blocks were compromised, then it would be easy to find out since all blocks are connected to each other. The blockchain is decentralized and cannot be corrupted, no single point of failure exists. The blockchain is where the actual voting takes place. The user's vote gets sent to one of the nodes on the system, and the node then adds the vote to the blockchain. The voting system will have a node in each district to ensure the system is decentralized.

2.2 Feasibility of e-Voting

Umut Can Çabuk, Eylül Adıgüzel, Enis Karaarslan, "A Survey on Feasibility and Suitability of Blockchain Techniques for the E-Voting Systems", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 7, Issue 3, March 2018

According to this paper [2] The blockchain technology owes its vast popularity to Bitcoin, the very first cryptocurrency. Shortly after, new cryptocurrencies arise. However, possible uses of blockchain is surely not limited to digital currencies. By April 2014, it is known that this chain is used for more than 80 different purposes under the name of sidechain.

The projects are classified according to the following characteristics:

- Proof of Identity: The identity of the user should be validated to use the system which is one of the most important security issues. This process involves using ID cards, or digital residence code or access to the official digital profile. The user must broadcast a proof of his/her identity when using Sovereign. The user should satisfy some criteria that can only be met by human judgment to avoid an AI from interfering with the process.

- User Interface: User interface is mostly web based in the e-voting platforms. Admin panel has included creation of vote, tracking process. User panel is more restricted, after verification it gives only permission to vote and results. When you signed up you can create a vote and you became an admin of that voting process or just vote as a user.

- Ledger: Ledger is a type of database that is shared, replicated, and synchronized among the members of a network. The ledger is distributed and records the transactions, such as the exchange of assets or data, among the participants in the network. This may show that it is open for advancement and is more prepared for the attacks.

- Consensus Protocol: The nodes, which share public ledgers need to use consensus algorithms to agree on a decision. The algorithms have to be functional, secure by design and also efficient. The consensus mechanisms are used by the nodes of the system to decide, mainly on who will get the right to update the block. This will ensure the integrity of the data recorded on the blockchain. Bitcoin and other mining based crypto coins use Proof-of-work (PoW) which depends on every node using their GPU power for that. This process is slow and uses extensive electricity, so alternatives are being developed.

- Programming Languages: Many programming languages can be used to develop blockchain applications and smart contracts. A smart contract is a computer protocol intended to digitally facilitate, verify, or enforce the negotiation or performance of a contract. Smart contracts allow the performance of credible transactions, which are trackable and irreversible, without intervention of any third parties (i.e. software or people). C++ and Python are widely used. Ethereum based solutions tend to use Solidity language for implementing smart contracts.

- Open Source: The users will trust the systems when they see the code.

2.3 The Future of e-Voting

Pavel Tarasov and Hitesh Tewari, "THE FUTURE OF E-VOTING", IADIS International Journal on Computer Science and Information Systems Vol. 12, No. 2, pp. 148-165

Electronic voting has been a topic of active debate, with significant number of people believing that electronic voting cannot be trusted enough to be used for significant elections due to uncertainty in the authenticity and integrity of the machines, and the votes that have been cast using them. On the other hand, people acknowledge that paper solutions are significantly outdated and can be subject to serious manipulation from a coercer. The emergence of blockchains has introduced a new way to construct secure systems which have less inherent security issues present within the systems. It is a belief that a successful voting system can be implemented using blockchains, or with a blockchain being one of the main elements present in a hybrid electronic voting scheme.

Zcash is a decentralized blockchain payment scheme, which aims to provide anonymity and privacy of transactions. One of the biggest differences between Zcash and Bitcoin is the proof-of-work system, where Zcash relies on zero-knowledge proofs. Zcash is an implementation of a concept called Zerocash which describes similar concepts to Zcash but the architecture behind Zcash is different.

2.4 Secure Voting System using Ethereum's Blockchain

Gaby G. Dagher, Praneeth Babu Marella, Matea Milojkovic and Jordan Mohler, "BroncoVote: Secure Voting System using Ethereum's Blockchain", 4th International Conference on Information Systems Security and Privacy

In this paper [4] BroncoVote, provides a secure and private e-voting system that is also easily accessible. BroncoVote is a university scale voting system that utilizes smart contracts in Ethereum and Paillier Homomorphic Encryption to achieve our goals. It provides voter privacy on all our ballots by encrypting every vote, homomorphically tallying, and revealing the vote count using Paillier cryptosystem decryption process. To maintain data integrity, all ballot and voting data is publicly available as part of the smart contracts or blockchain in our system.

A certain type of difficulty faced in the development of BroncoVote is support for cryptography, the maximum data value Solidity is unsigned int of 256 bit. The majority of encryption protocols require much larger integer numbers than are supported in Solidity.

Proof of concept system for BroncoVote that utilized the Ethereum blockchain and Paillier homomorphic encryption. Our implementation was tested on the Ethereum testnet network with different types and sizes of ballots. BroncoVote used the smart contracts in Ethereum blockchain to keep a record of every user in our system as well as all the ballots and the information regarding them. We also utilized the smart contracts to achieve access control. We integrated Paillier homomorphic encryption into our system to preserve voter privacy. With the deployment of our system on the testnet for experiments we showed that our system can easily be deployed and setup.

2.5 Concept of Blockchain Voting in Real Life

Ahmed Ben Ayed, "A CONCEPTUAL SECURE BLOCKCHAIN- BASED ELECTRONIC VOTING SYSTEM", International Journal of Network Security & Its Applications (IJNSA) Vol.9, No.3, May 2017

This paper [5] talks about how their e-Voting solution will include four main requirements that can be illustrated as shown below:

- Authentication: Only people already registered to vote can cast a vote. Our system will not support a registration process. Registration usually requires verification of certain information and documents to comply with current laws, which could not be done online in a secure manner. Therefore, the system should be able to verify voters' identities against a previously verified database, and then let them vote only once.

- Anonymity: The e-Voting system should not allow any links between voters' identities and ballots. The voter has to remain anonymous during and after the election.

- Accuracy: Votes must be accurate; every vote should be counted, and can't be changed, duplicated or removed.

- Verifiability: The system should be verifiable to make sure all votes are counted correctly. Beside the main requirement, our solution supports mobility, flexibility, and efficiency. However, we will limit this paper's discussion to the four main requirements.

2.6 Types of Blockchain Frameworks

Friðrik Þ. Hjálmarsson, Gunnlaugur K. Hreiðarsso, "Blockchain-Based e-Voting System", 2018 IEEE 11th International Conference on Cloud Computing (CLOUD)

This paper [6] shows comparison between the three blockchain frameworks that we consider for implementing and deploying our election smart contracts. Those are Exonum, Quorum and Geth.

1) Exonum: Looking at the Exonum blockchain, it is robust end to end with its full implementation done with the programming language Rust. Exonum is built for private blockchains. It has customized Byzantine algorithm that is used to achieve consensus in the network. With that consensus algorithm, Exonum can support up to 5000 transactions per second. Unfortunately, the limitation of the framework is that Rust is the only programming language in the current version, which limits the developers to the constructs available in that language. To solve this limitation, Exonum is planning to introduce Javabindings and platform-independent interface description to make Exonum more developer-friendly in the near future.

2) Quorum: Is an Ethereum-based distributed ledger protocol with transaction/contract privacy and new consensus mechanisms. It's a Geth fork and is updated in line with Geth releases. Quorum changed up the consensus mechanism and aimed more towards consortium chain-based consensus algorithms. Using this consensus allows it to support from dozens to hundreds of transactions per second.

3) Geth: Go-Ethereum or Geth is one of three original implementations of the Ethereum protocol and it runs smart contract applications exactly as programmed without possibility of downtime, censorship, fraud or third-party interference. This framework supports development beyond the Geth protocol, and is the most developer-friendly framework of the frameworks we evaluated. The transaction per second (transaction rate) is dependent on whether the blockchain is implemented as a

public or private network. Because of these capabilities, Geth was the framework we chose to base our work on, any similar blockchain framework with the same capabilities as Geth should be considered for such systems.

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATION

3.1 Functional requirements

1. Mobility: The voter should not be restricted to cast his ballot at a single poll-site at his home precinct.

- Realistic: He shall be able to vote from any poll-site within the nation.
- Unrealistic/Expensive: He shall be able to vote from any county-controlled kiosk (situated at public places such as banks, shopping malls, etc.) within the nation. (Unrealistic because of logistical and cost issues).
- Infeasible: He shall be able to vote from virtually anywhere using an Internet connection. (Infeasible both for technical security issues as well as social science issues).

2. Convenience: The system shall allow the voters to cast their votes quickly, in one session, and should not require many special skills or intimidate the voter (to ensure Equality of Access to Voters).

3. User-Interface: The system shall provide an easy-to-use user-interface. Also, it shall not disadvantage any candidate while displaying the choices (e.g., by requiring the user to scroll down to see the last few choices).

4. Transparency: Voters should be able to possess a general knowledge and understanding of the voting process.

5. Flexibility: The system shall be flexible in that it allows a variety of ballot question formats including open-ended questions (e.g. Write-in candidates and survey questions).

6. Support for Disabled Voters: The system shall cater to the needs of physically challenged voters (e.g. blind voters).

7. Accuracy: The system shall record and count all the votes and shall do so correctly.

8. Eligibility: Only authorized voters, who are registered, should be able to vote.

9. Uniqueness: No voter should be able to vote more than once.

10. Auditability: It should be possible to verify that all votes have been correctly accounted for in the final election tally, and there should be reliable and demonstrably authentic election records, in terms of physical, permanent audit trail (which should not reveal the user's identity in any manner).

11. Voter Confirmation: The voter shall be able to confirm clearly how his vote is being cast, and shall be given a chance to modify his vote before he commits it.

12. To issue Receipt or not? The system may issue a receipt to the voter if and only if it can be ensured that vote-coercion and vote-selling are prevented, so that he may verify his vote at any time and also contend, if necessary.

13. No Over-voting: The voter shall be prevented from choosing more than one candidate / answer.

14. Under-voting: The voter may receive a warning of not voting, but the system must not prevent undervoting.

15. Provisional Ballots: The voter shall be able to vote with a provisional (electronic) ballot if he has some registration problems, which could be counted if verified by the authorities later.

16. Documentation and Assurance: The design, implementation, and testing procedures must be well documented so that the voter-confidence in the election process is ensured.

17. Cost-effectiveness: Election systems should be affordable and efficient.

3.2 Non-Functional Requirements

Non-functional requirements are requirements that are not specifically concerned with the functionality of a system. They normally place restrictions on the product being developed and the development process [8]. Non-functional requirements may be regarded as parameters of functionality in that they determine how quickly, how accurately, how reliably, how securely, etc., functions must operate [2]. Some of the Online Voting System's non-functional requirements are as follows:

- Response and net processing time must be acceptable by user and by application.
- Defects in the local voting database file must be less than a very small positive value, ε, according to the six sigma representation [3].
- Defects contained in the collection server must be less than a very small positive value, ε, according to the six sigma representation.
- Defects in the master/server database must be less than a very small positive value, ε, according to the six sigma representation.
- Number of collection failures per voting process must be at six sigma, or better. Taking the entire state of Michigan, with a population of about 9.9M citizens (if all voted), as an example, this works out to be 33 errors.
- When checking the database for errors, a 100% scan of the data is required, rather than selecting a sample set.
- The system must be working at 100% peak efficiency during the voting process.
- Transfer of existing and future data to a Voting Management Data Center must be granted.
- The system should be allow adding more voters to allow a greater connectivity rate.
- A process must be devised to support normal precinct business hours.
- Due to the shortness of the voting timeframe, the system should support response time for addressing severe issues in less than 5 minutes.
- The system should provide documentation to inform users of system functionality and any change to the system.
- The system should provide friendly graphical Interface to ensure ease of use when end users utilize system functionality

3.3 Feasibility Analysis

- Disintermediation: The transactions are not verified by a single central gatekeeper and this may reduce the costs of building an infrastructure and its maintenance, and there may be some performance gains.
- Transaction interaction: Complicated and interconnected transactions can be implemented by using smart contracts. Blockchain structure provides an easy and modular base for public key infrastructures and blind signatures.
- Auditability: Every record in blockchain keeps who are involved in the transactions, as well as the type, amount the content.
- Full trust, such that the election should not be under the control of anybody.
- Remote participation, Voting can be done from any booth
- Immutable and non-manipulatable records of votes.

3.4 Hardware Requirements:

Processor: Pentium IV

Memory: 1 GB RAM

Hard Disk Drive: 2 GB

3.5 Software Requirements:

Platform: Windows/Linux/macOS

Language used: JavaScript

Back end: Docker, Hyperledger Fabric, NodeJS, ExpressJS, PassportJS, MongoDB, Mongoose

Technologies used: Blockchain, Smart Contracts

Design tools: HTML, CSS, Bootstrap

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

This chapter talks about the system design used for this project along with the UML. In this case we have used a use case diagram to represent our project. We also discuss the architecture of the system.

4.1 System Design

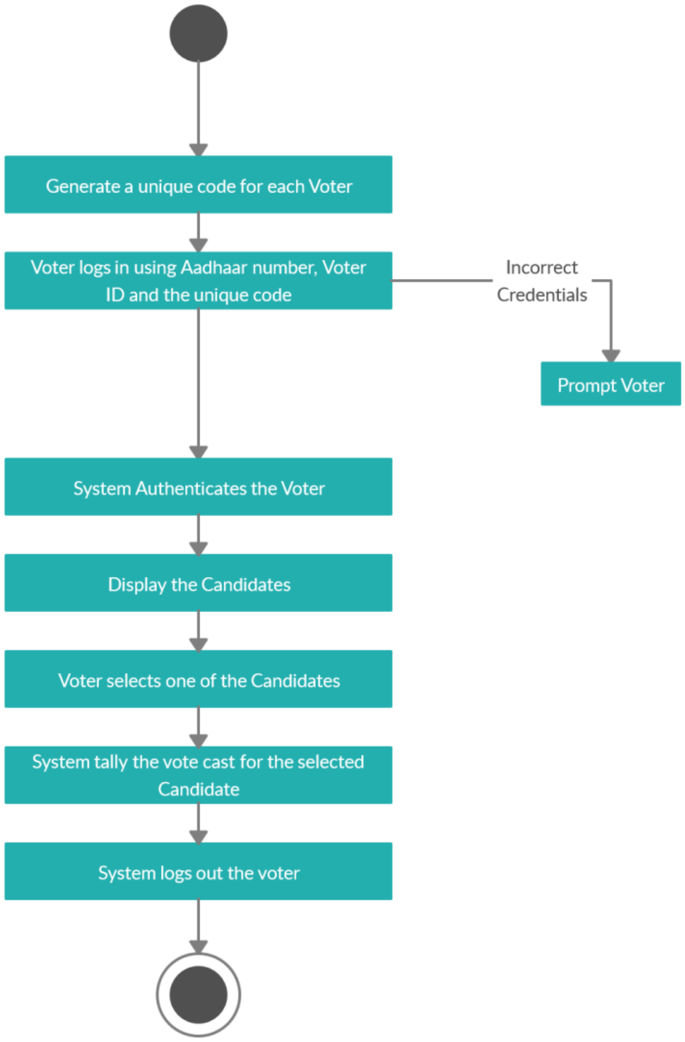

Generate a unique code for each Voter
Voter logs in using Aadhaar number, Voter ID and the unique code
Incorrect Credentials
Prompt Voter
System Authenticates the Voter
Display the Candidates
Voter selects one of the Candidates
System tally the vote cast for the selected Candidate
System logs out the voter

Fig 4.1 Flowchart/ Activity diagram

From figure 4.1, we can see that each voter has a unique Aadhaar and voter ID. Using this they log into the voting system. If a voter who has already voted enters his credentials, the system shows a warning for double voting. If the voter is legitimate, they are authorized to cast the vote. Once logged in, the voter is allowed to vote for a party of his choice from the options provided. When the vote is cast, the voter can submit and the vote is now added to the blockchain of votes. The voter is logged out of the system after this process.

To view the tally of the votes, the admin must login through the admin login website and view the information on the dashboard.

4.2 Use Case Diagram

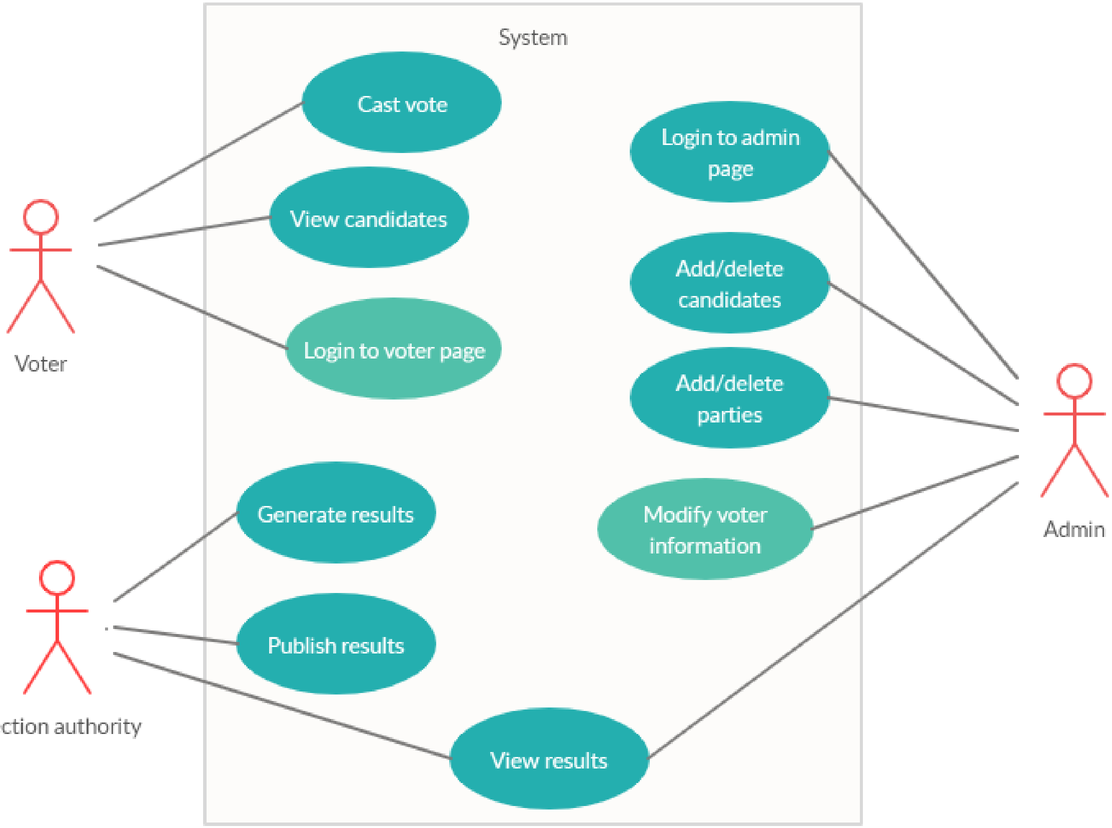


Fig 4.2 Use Case Diagram

Here in figure 4.2, we have three actors – the voter, the admin and the election authority. Each of them is given special access as shown in the diagram. The voter gets the least access where they are only allowed to login and cast vote. The election authority is only allowed to view and publish the results of the election. They are not allowed to tamper/modify any information. Whereas the admin gets the most privilege. They are allowed to add/delete/modify the candidates, parties and voter information.

4.3 Architecture

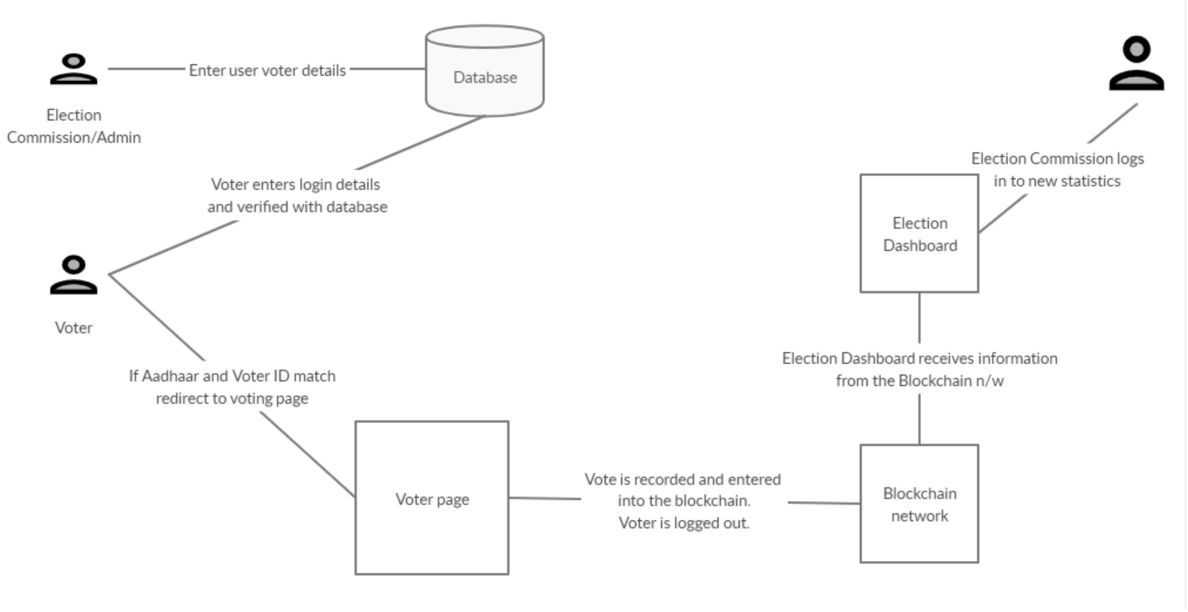

Enter user voter details
Database
Election Commission/Admin
Voter enters login details and verified with database
Voter
If Aadhaar and Voter ID match redirect to voting page
Voter page
Vote is recorded and entered into the blockchain. Voter is logged out.
Blockchain network
Election Dashboard receives information from the Blockchain n/w
Election Dashboard
Election Commission logs in to new statistics

Fig 4.3 Architecture

Figure 4.3 illustrates how the system works. The admin first creates an account for the voter using Aadhaar and Voter ID details, and the login information is stored in the database. Once registered, the voters use this information (Aadhaar and Voter ID) to log into the voting site at a polling booth location provided to them and cast their vote. After vote is cast, voter is logged out to prompt next voter. The vote is recorded and entered into the blockchain network. The Election Commission are the only ones authorized to log in to the Election Dashboard. This provides statistics of the election such as no. of male and female voters, no. of voters per state, voter turnout and so on.

CHAPTER 5

IMPLEMENTATION

5.1 Chaincode

This snippet of code is the backbone of the system. It is used to add, query votes and search for distinct parties, cities and states.

```
'use strict';
const {
    Contract
} = require('fabric-contract-api');

var height = 0;

class VotingContract extends Contract {

    async init(ctx) {
        console.info("Instantiated Election Chaincode...");
    }

    async initLedger(ctx) {

        /*let voteInfo = {
          voterId: 0,
          aadhaarNumber: 0,
          party: 0,
          city: 0,
          state: 0,
          timestamp: 0,
          votingCenterId: 0,
          height: 0,
          gender: 0,
        };
        await ctx.stub.putState('0000',
Buffer.from(JSON.stringify(voteInfo)));*/
        console.info("Instantiated Election Chaincode...");

    }

    async castVote(ctx, voterId, aadhaarNumber, party, city,
state, votingCenterId, gender) {

        let votesAsBytes = await ctx.stub.getState(voterId);
        if (!votesAsBytes || votesAsBytes.toString().length <= 0)
{
            let timeStamp = await ctx.stub.getTxTimestamp();
            height = height + 1;
            let voteInfo = {
                voterId: voterId,
                aadhaarNumber: aadhaarNumber,
                party: party,
                city: city,
                state: state,
                timestamp: timeStamp,
                votingCenterId: votingCenterId,
                height: height,
                gender: gender,
            };
            await ctx.stub.putState(voterId,
Buffer.from(JSON.stringify(voteInfo)));
            console.log('Vote added To the ledger
Succesfully..');
        } else {
            console.log('Double voting not allowed');
        }
    }

    async searchVote(ctx, voterId) {
        let votesAsBytes = await ctx.stub.getState(voterId);
        if (!votesAsBytes || votesAsBytes.toString().length <= 0)
{
            throw new Error('Voter with this Id does not exist');
        }
        let voteInfo = JSON.parse(votesAsBytes.toString());
        return JSON.stringify(voteInfo);
    }

    async queryAllVotes(ctx) {
        const startKey = '';
        const endKey = '';

        const iterator = await ctx.stub.getStateByRange(startKey,
endKey);
        const allResults = [];

        while (true) {
            const res = await iterator.next();
            if (res.value && res.value.value.toString()) {
                console.log(res.value.value.toString('utf8'));

                const key = res.value.key;
                console.log("Key => " + key);

                let Record;
                try {
                    Record =
JSON.parse(res.value.value.toString('utf8'));
                } catch (err) {
                    console.log(err);
                    Record = res.value.value.toString('utf8');
                }
                allResults.push(Record);
            }
            if (res.done) {
                console.log("Reached end of blocks...");
                await iterator.close();
                console.info(allResults);
                return allResults;
            }
        }
    }

    async getDistinctParties(ctx) {
        var allVotes = await this.queryAllVotes(ctx);
        const result = [];
        const map = new Map();
        for (const item of allVotes) {
            if (!map.has(item.party)) {
                map.set(item.party, true); // set any value to
Map
                result.push(item.party);
            }
        }
        return result;
    }

    async getPartyVoteSplit(ctx) {
        var allVotes = await this.queryAllVotes(ctx);
        var distinctParties = await this.getDistinctParties(ctx)
        var result = {}

        for (const party of distinctParties) {
            result[party] = 0;
        }

        for (const vote of allVotes) {
            result[vote['party']] = result[vote['party']] + 1;
        }
        return result;
    }

    async description(ctx) {
        var result = {};
        var totalVotes = 0;
        var totalMaleVotes = 0;
        var totalOtherVotes = 0;
        var totalFemaleVotes = 0;
        var distinctParties = [];
        var cityVotes = {};
        var stateVotes = {};
        var malePartyVotes = {};
        var femalePartyVotes = {};
        var totalPartyVotes = {};
        var votesPartyState = {};

        const startKey = '';
        const endKey = '';

        const iterator = await ctx.stub.getStateByRange(startKey,
endKey);
        while (true) {
            const res = await iterator.next();
            if (res.value && res.value.value.toString()) {
                console.log(res.value.value.toString('utf8'));
                let Record;
                try {

                    Record =
JSON.parse(res.value.value.toString('utf8'));
                    totalVotes = totalVotes + 1;

                    //Gender vote count
                    if (Record['gender'] === 'MALE') {
                        totalMaleVotes = totalMaleVotes + 1;
                    }
                    if (Record['gender'] === 'FEMALE') {
                        totalFemaleVotes = totalFemaleVotes + 1;
                    }
                    if (Record['gender'] === 'OTHER') {
                        totalOtherVotes = totalOtherVotes + 1;
                    }

                    //Distinct Parties
                    if
(!distinctParties.includes(Record['party'])) {
                        distinctParties.push(Record['party']);
                    }

                    //City Votes
                    if
(Object.keys(cityVotes).includes(Record['city'])) {
                        cityVotes[Record['city']] =
cityVotes[Record['city']] + 1;
                    } else {
                        cityVotes[Record['city']] = 1;
                    }

                    //State allVotes
                    if
(Object.keys(stateVotes).includes(Record['state'])) {
                        stateVotes[Record['state']] =
stateVotes[Record['state']] + 1;
                    } else {
                        stateVotes[Record['state']] = 1;
                    }

                    //Total party votes
                    if
(Object.keys(totalPartyVotes).includes(Record['party'])) {
                        totalPartyVotes[Record['party']] =
totalPartyVotes[Record['party']] + 1;
                    } else {
                        totalPartyVotes[Record['party']] = 1;
                    }

                    //Votes per party per state
                    var party = {};
                    if
(Object.keys(party).includes(Record['party']) &&
Object.keys(state).includes(Record['state'])) {
                        party[Record['party']] =
party[Record['party']] + 1;
                        votesPartyState[Record['state']] = party;
                    } else {
                        party[Record['party']] = 1;
                        votesPartyState[Record['state']] = party;
                    }
                    var party = {};


                } catch (err) {
                    console.log(err);
                    Record = res.value.value.toString('utf8');
                }


            }
            if (res.done) {
                await iterator.close();
                result['totalVotes'] = totalVotes;
                result['totalMaleVotes'] = totalMaleVotes;
                result['totalOtherVotes'] = totalOtherVotes;
                result['totalFemaleVotes'] = totalFemaleVotes;
                result['distinctParties'] = distinctParties;
                result['cityVotes'] = cityVotes;
                result['stateVotes'] = stateVotes;
                result['totalPartyVotes'] = totalPartyVotes;
                result['votesPartyState'] = votesPartyState;
                return result;
            }
        }

    }

}

module.exports = VotingContract;
```

5.2 Server to run voter and admin front end

This snippet of code is used to run the server on both voter and admin end. It uses key elements such as PassportJS, ExpressJS, Mongoose and EJS.

```
require('dotenv').config();
const bodyParser = require('body-parser');
const express = require('express');
const app = express();
const ejs = require('ejs');
const mongoose = require('mongoose');
const encrypt = require('mongoose-encryption');
const session = require('express-session');
const passport = require('passport');
const passportLocalMongoose = require('passport-local-mongoose');
const Schema = mongoose.Schema;
var generator = require('generate-password');
const {
  exec
} = require("child_process");
//const Instascan = require('instascan');

app.set('view engine', 'ejs');
app.use(express.static('public'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({
  extended: true
}));

//----------DELETE BLOCKCHAIN USERS [DEBUG]----------
exec("rm -r wallet", (error, stdout, stderr) => {
  if (error) {
    console.log(`error: ${error.message}`);
    return;
  }
  if (stderr) {
    console.log(`stderr: ${stderr}`);
    return;
  }
  console.log(`stdout: ${stdout}`);
});
//----------END----------


//----------SESSION INFORMATION----------
//app.set('trust proxy', 1)
app.use(session({
  secret: 'keyboard meow',
  resave: false,
  saveUninitialized: false,
  /*cookie: {
    secure: true
  }*/
}))

app.use(passport.initialize());
app.use(passport.session());
//----------END----------

//----------HYPERLEDGER OBJECTS----------
const enrollAdmin = require('../server/enrollAdmin');
const registerUser = require('../server/registerUser');
const query = require('../server/query');
const invoke = require('../server/invoke');
//----------END----------

//----------VOTING PROCESS SETUP----------
async function enrollBlockchainAdmin() {
  return await enrollAdmin.enrollAdmin();
}

async function enrollBlockchainUser(username) {
  return await registerUser.registerUser(username);
}

enrollBlockchainAdmin().then(function(result) {
  console.log('enrollBlockchainUser :', result);
  enrollBlockchainUser('votingUser').then(function(result) {
    console.log(result);
  });
});

const partyLogos = {
  BJP : 'https://www.freepnglogos.com/uploads/bjp-logo-
png/bharatiya-janata-party-logos-download-3.png',
  INC : 'https://banner2.cleanpng.com/20180526/bz/kisspng-
dehradun-indian-national-congress-bharatiya-janata-
5b0914d88dd800.446294781527321816581.jpg',
  AAP : 'https://pngimage.net/wp-content/uploads/2018/05/aap-
logo-png-1.png',
  DMK :
'https://upload.wikimedia.org/wikipedia/commons/0/0e/Risingsun_su
rya.png',
};


async function invokeContract(username, voterId, aadhaarNumber,
party, city, state, votingCenterId, gender) {
  return await invoke.invoke(username, voterId, aadhaarNumber,
party, city, state, votingCenterId, gender);
}

//----------END----------

//----------CONNECT TO MONGODB----------
mongoose.connect(process.env.DB_ADDRESS, {
  useNewUrlParser: true,
  useUnifiedTopology: true
}, function(err) {
  if (err) {
    console.log(err);
  } else {
    console.log("Established connection with DB :)");
  }
});
//-----------END-----------

const voterSchema = new Schema({
  username: String,
  voterId: String,
  password: String,
  hasVoted: String,
});
```

```
voterSchema.plugin(passportLocalMongoose);

const Voter = mongoose.model('voter_list', voterSchema);
passport.use(Voter.createStrategy());
passport.serializeUser(Voter.serializeUser());
passport.deserializeUser(Voter.deserializeUser());

//----------ROUTES----------

app.get('/registervoter', function(req, res) {
  res.render('registervoter');
});

app.post('/registervoter', function(req, res) {
  Voter.findOne({
    username: req.body.username,
  }, function(err, person) {
    if (err) {
      console.log(err);
    } else {
      if (person) {
        console.log(person,'already registered');
        res.redirect('/registerVoter');
      } else {
        Voter.register({username: req.body.username, hasVoted:
false, voterId: req.body.password}, req.body.password,
function(err, voter) {
          if (err) {
            console.log(err);
            res.redirect('/registervoter');
          } else {
            console.log('Voter', voter, 'registered
successfully');
            res.redirect('/registervoter');
          }
        });
      }
    }
  });
});


app.get('/', function(req, res) {
  req.logout();
  res.redirect('/voterlogin');
});

app.get('/voterlogin', function(req, res) {
  req.logout();
  res.render('voterlogin1', {
    showAlerts: false,
    //Instascan: Instascan,
  });
});

app.get('/castvote', function(req, res) {
  if (req.isAuthenticated()) {
    res.render('castvote', {
      partyLogos: partyLogos,
      aadhaarNumber: req.user.username,
      voterId: req.user.voterId,
    });
  } else {
    res.redirect('/voterlogin')
  }
});

app.post('/voterlogin', function(req, res) {
  Voter.findOne({username: req.body.username}, function(err,
voter){
    if(err){
      console.log(err);
    } else {
      if(voter){
        if(voter['hasVoted'] == 'true'){
          //DOUBLE VOTING CHECK 1
          res.render('voterlogin1', {
            showAlerts: true,
            validCredentials: true,
            doubleVotingAttempt: true,
            //Instascan: Instascan,
          });
        } else {
          const voter = new Voter({
            username: req.body.username,
            password: req.body.password,
          });
          req.login(voter, function(err) {
            if (err) {
              console.log(err);
              res.render('voterlogin1', {
                showAlerts: true,
                validCredentials: false,
                doubleVotingAttempt: false,
                //Instascan: Instascan,
              });
            } else {
              console.log('success');
              passport.authenticate('local')(req, res, function()
{
                res.redirect('/castvote');
              });
            }
          });
        }
      }
      else {
        res.render('voterlogin1', {
          showAlerts: true,
          validCredentials: false,
          doubleVotingAttempt: false,
          //Instascan: Instascan,
        });
      }
    }
  })
});

app.post('/castvote', function(req, res){
  //console.log('option',req.body.party);
  //console.log('aadhaar',req.user.username);
  //console.log('voterid', req.user.voterId);
  //const username = req.user.username;
  //const voterId = req.user.voterId;

  invokeContract('votingUser', req.user.voterId,
req.user.username, req.body.party, 'BENGALURU', 'KA', '0001',
'MALE').then(function(result){
    if(result){
      console.log('Vote successfully added to ledger');
      Voter.updateOne({'username': req.user.username}, {hasVoted:
true}, {upsert: true}, function(err, doc){
        if(err){
          console.log(err);
          res.redirect('/castvote');
        } else {
          console.log(doc);
          res.render('displayvote',{
            vote: req.body.party,
            aadhaarNumber: req.user.username,
            voterId: req.user.voterId,
          });
          //req.logout();
          //res.redirect('/');
        }
      });
    }
    else {
      console.log('Hyperledger error');
      res.redirect('/castvote');
    }
  });
});


app.get('/logout', function(req, res) {
  req.logout();
  res.redirect('/');
});
//-----------END-----------

//-----------START SERVER-----------
app.listen(5000, '10.0.0.9', function() {
  console.log("Voting server active on port 5000");
});
//-----------END-----------
```

5.3 Voter Login

```
<!DOCTYPE html>
<html>

<head>
  <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
  <!-- Include all compiled plugins (below), or include
individual files as needed -->
  <script src="../js/bootstrap.min.js"></script>
  <link
href="../bower_components/select2/dist/css/select2.min.css"
rel="stylesheet">
  <link href="../css/main.css?version=4.4.0" rel="stylesheet">
  <script
src="../bower_components/jquery/dist/jquery.min.js"></script>
  <script type="text/javascript"
src="https://rawgit.com/schmich/instascan-
builds/master/instascan.min.js"></script>
  <%if(showAlerts){%>
  <%if(!validCredentials){%>
  <div class="alert alert-danger alert-dismissible fade show"
role="alert">
    <button aria-label="Close" class="close" data-dismiss="alert"
type="button"><span aria-hidden="true">
&times;</span></button><strong>Error : </strong>Invalid
credentials
  </div>
  <%}%>
  <%if(doubleVotingAttempt){%>
  <div class="alert alert-danger alert-dismissible fade show"
role="alert">
    <button aria-label="Close" class="close" data-dismiss="alert"
type="button"><span aria-hidden="true">
&times;</span></button><strong>Error : </strong>Double voting
attempt detected
  </div>
  <%}}%>
  <title>Voter Login</title>
  <meta charset="utf-8">
  <meta content="ie=edge" http-equiv="x-ua-compatible">
  <meta content="width=device-width, initial-scale=1"
name="viewport">
  <link href="../favicon.png" rel="shortcut icon">
  <link href="../apple-touch-icon.png" rel="apple-touch-icon">
  <link
href="https://fonts.googleapis.com/css?family=Rubik:300,400,500"
rel="stylesheet" type="text/css">
  <link href="../css/main.css?version=4.4.0" rel="stylesheet">
</head>
<script>
  window.setTimeout(function() {
    $(".alert-danger").fadeTo(500, 0).slideUp(500, function() {
      $(this).remove();
    });
  }, 5000);
</script>

<body>
  <div class="all-wrapper menu-side with-pattern">
    <div class="auth-box-w wider">
      <div class="logo-w">
        <a><img alt="" src="img/logo-big.png"></a>
        <a class="drdo-logo"><img src="img/drdo-logo.png"
height="70px" width="70px"></a>
      </div>
      <h4 class="auth-header">
        Enter Credentials
        <button class="mr-2 mb-2 btn btn-primary" data-
target="#aadhaarScanModal" data-toggle="modal" type="button">Scan
Aadhaar Card</button>
      </h4>
      <div aria-hidden="true" class="onboarding-modal modal fade
animated" id="aadhaarScanModal" role="dialog" tabindex="-1">
        <div class="modal-dialog modal-centered" role="document">
          <div class="modal-content text-center">
            <button aria-label="Close" class="close" data-
dismiss="modal" type="button"><span class="os-icon os-icon-
close"></span></button>
            <div class="onboarding-media">
              <img alt="" src="img/bigicon1.png" width="120px">
            </div>
            <div class="onboarding-content with-gradient">
              <h4 class="onboarding-title">

              </h4>
              <div class="onboarding-text">
                Hold QR code at camera
              </div>
              <video id="preview"></video>
              <script type="text/javascript">
                let scanner = new Instascan.Scanner({
                  video: document.getElementById('preview')
                });
                scanner.addListener('scan', function(content) {
                  alert(content);
                });

Instascan.Camera.getCameras().then(function(cameras) {
                  if (cameras.length > 0) {
                    scanner.start(cameras[0]);
                  } else {
                    console.error('No cameras found.');
                  }
                }).catch(function(e) {
                  console.error(e);
                });
              </script>
            </div>
          </div>
        </div>
      </div>
      <form class="form" name="form" method="post"
action="/voterlogin" autocomplete="off">
        <div class="row">
          <div class="col-sm-6">
            <div class="form-group">
              <label for="">Aadhaar Number</label>
              <input name="username" id="username" class="form-
control" autocomplete="false" placeholder="" type="password"
onchange=validate()>
              <div class="pre-icon os-icon os-icon-
fingerprint"></div>
            </div>
          </div>
          <div class="col-sm-6">
            <div class="form-group">
              <label for="">Voter ID</label>
              <input name="password" id="password" class="form-
control" autocomplete="false" placeholder="" type="password"
onchange=validate()>
            </div>
          </div>
        </div>
        <div class="buttons-w">
          <button class="btn btn-primary" type="submit"
disabled>Submit</button>
        </div>
      </form>
    </div>
  </div>
  <script
src="../bower_components/popper.js/dist/umd/popper.min.js"></scri
pt>
  <script src="../bower_components/bootstrap-
validator/dist/validator.min.js"></script>
  <script src="../bower_components/perfect-scrollbar/js/perfect-
scrollbar.jquery.min.js"></script>
  <script
src="../bower_components/bootstrap/js/dist/util.js"></script>
  <script
src="../bower_components/bootstrap/js/dist/alert.js"></script>
  <script
src="../bower_components/bootstrap/js/dist/button.js"></script>
  <script
src="../bower_components/bootstrap/js/dist/carousel.js"></script>
  <script
src="../bower_components/bootstrap/js/dist/collapse.js"></script>
  <script
src="../bower_components/bootstrap/js/dist/dropdown.js"></script>
  <script
src="../bower_components/bootstrap/js/dist/modal.js"></script>
  <script type="module" src="js/main.js?version=4.4.0"></script>
</body>
<script type="text/javascript">
  $("form").keydown(function() {
    validate();
  });

  function validate() {
    if ($('#username').val().length > 0 &&
      $('#password').val().length > 0) {
      $("button[type=submit]").prop("disabled", false);
    } else {
      $("button[type=submit]").prop("disabled", true);
    }
  }
</script>

</html>
```

5.4 Register Voter

```
<!DOCTYPE html>
<html>
  <head>
    <title>Voter Registration Portal</title>
    <meta charset="utf-8">
    <meta content="ie=edge" http-equiv="x-ua-compatible">
    <meta content="width=device-width, initial-scale=1"
name="viewport">
    <link href="../favicon.png" rel="shortcut icon">
    <link href="../apple-touch-icon.png" rel="apple-touch-icon">
    <link
href="https://fonts.googleapis.com/css?family=Rubik:300,400,500"
rel="stylesheet" type="text/css">
    <link href="../css/main.css?version=4.4.0" rel="stylesheet">
  </head>
  <body class="auth-wrapper">
    <div class="all-wrapper menu-side with-pattern">
      <div class="auth-box-w">
        <div class="logo-w">
          <a><img alt="" src="/img/logo-big.png"></a>
          <a class="drdo-logo"><img src="/img/drdo-logo.png"
height="70px" width="70px"></a>
        </div>
        <h4 class="auth-header">
          Voter Registration
        </h4>
        <form class="form" action="/registervoter" method="post"
autocomplete="off">
          <div class="form-group">
            <label for="">Aadhaar Number</label>
            <input name="username" class="form-control"
placeholder="" type="text">
            <div class="pre-icon os-icon os-icon-user-male-
circle"></div>
          </div>
          <div class="form-group">
            <label for="">Voter ID Number</label>
            <input name="password" class="form-control" 
placeholder="" type="password">
            <div class="pre-icon os-icon os-icon-
fingerprint"></div>
          </div>
          <div class="buttons-w">
            <button class="btn btn-primary" 
type="submit">Register</button>
          </div>
        </form>
      </div>
    </div>
  </body>
</html>
```

5.5 Cast Vote

```
<!DOCTYPE html>
<html>

<head>
  <title>Cast Vote</title>
  <meta charset="utf-8">
  <meta content="ie=edge" http-equiv="x-ua-compatible">
  <link href="favicon.png" rel="shortcut icon">
  <link href="apple-touch-icon.png" rel="apple-touch-icon">
  <link 
href="https://fonts.googleapis.com/css?family=Rubik:300,400,500" 
rel="stylesheet" type="text/css">
  <link href="css/main.css?version=4.4.0" rel="stylesheet">
</head>

<body class="menu-position-side menu-side-left full-screen with-
content-panel">
  <div class="all-wrapper with-side-panel solid-bg-all">
    <div class="layout-w">
      <div class="content-w">
        <div class="content-i">
          <div class="content-box">
            <div class="row">
              <div class="col-sm-12">
                <div class="element-wrapper">
                  <div class="element-box">
                    <form id="voteForm" method="post" 
action="/castvote" autocomplete="off">
                      <h5 class="form-header">
                        Cast Vote
                        <h6>Aadhaar Number : 
<%=aadhaarNumber%></h6>
                        <h6>Voter ID : <%=voterId%></h6>
                      </h5>
                      <div class="form-desc">
                        Select the party that you wish to vote 
for
```

```
                      </div>
                      <div class="form-group">
                        <%for(party of 
Object.keys(partyLogos)){%>
                        <div class="row">
                          <div class="col-sm-12">
                            <div class="op">
                              <input type="radio" name="party" 
value=<%=party%>>
                              <img src=<%=partyLogos[party]%> 
height="100px" width="100px">
                              <label style="font-
size:200%;"><%=party%></label><br>
                            </div>
                          </div>
                        </div>
                        <%}%>
          </div>
          <div class="form-check">
            <label class="form-check-label"><input class="form-
check-input" type="checkbox" id="terms_and_conditions" 
onclick="terms_changed(this)">Confirm vote</label>
          </div>
          <div class="form-buttons-w">
            <button class="btn btn-primary" id="submit_button" 
type="submit" disabled> Submit</button>
          </div>
        </form>
      </div>
    </div>
  </div>
</div>
            </div>
          </div>
        </div>
      </div>
      <div class="display-type"></div>
    </div>
    <script src="js/demo_customizer.js?version=4.4.0"></script>
    <script src="js/main.js?version=4.4.0"></script>
    <script type="text/javascript">
    function terms_changed(termsCheckBox){
      //If the checkbox has been checked
      if(termsCheckBox.checked){
          //Set the disabled property to FALSE and enable the 
button.
          document.getElementById("submit_button").disabled = 
false;
      } else{
          //Otherwise, disable the submit button.
          document.getElementById("submit_button").disabled = 
true;
      }
    }
    </script>
  </body>
```

```
</html>
```

5.6 Display Vote

```
<!DOCTYPE html>

<html>

<head>
  <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.m
in.js"></script>
  <!-- Include all compiled plugins (below), or include
individual files as needed -->
  <script src="js/bootstrap.min.js"></script>
</head>

<body style="background-color: #222533; padding: 20px; font-
family: font-size: 14px; line-height: 1.43; font-family:
&quot;Helvetica Neue&quot;, &quot;Segoe UI&quot;, Helvetica,
Arial, sans-serif;">
  <div style="max-width: 600px; margin: 0px auto; background-
color: #fff; box-shadow: 0px 20px 50px rgba(0,0,0,0.05);">
    <table style="width: 100%;">
      <tr>
        <td style="background-color: #fff;">
          <img alt="" src="img/logo.png" style="width: 70px;
padding: 20px">
          <img alt="" src="img/drdo-logo.png" style="width: 70px;
padding: 10px">
        </td>
      </tr>
    </table>
    <div style="padding: 60px 70px; border-top: 1px solid
rgba(0,0,0,0.05);">
      <h1 style="margin-top: 0px;">
        Your vote has been successfully recorded
      </h1>
      <div style="color: #636363; font-size: 14px;">
        <p>
          You will be redirected in 5 seconds
        </p>
      </div>
      <div style="border: 2px solid #4B72FA; padding: 40px;
margin: 40px 0px;">
        <h4 style="margin-bottom: 20px; margin-top: 0px; font-
size: 18px; display: inline-block; border-bottom: 1px dotted
#111;">
          Your vote information
        </h4>
        <table style="width: 100%;">
          <tr>
```

```
            <td style="padding-left: 30px; border-bottom: 1px
solid rgba(0,0,0,0.05);">
              <div style="font-weight: bold; color: #636363;
font-size: 16px;">
                <h4>Aadhaar Number : <span
class="greenText"><%=aadhaarNumber%></span></h4>
                <h4>Voter ID : <span
class="greenText"><%=voterId%></span></h4>
                <h4>Vote : <span
class="greenText"><%=vote%></span></h4>
              </div>
            </td>
          </tr>
        </table>
      </div>
    </div>
    <script>
      setTimeout(function() {
        // after 2 seconds
        window.location = "/";
      }, 5000)
    </script>
</body>

</html>
```

CHAPTER 6

RESULTS AND DISCUSSION

The system developed allows the admin to upload all the voter information in encrypted form into a database.

Once this is done, the voter will be allowed to log in and case vote.

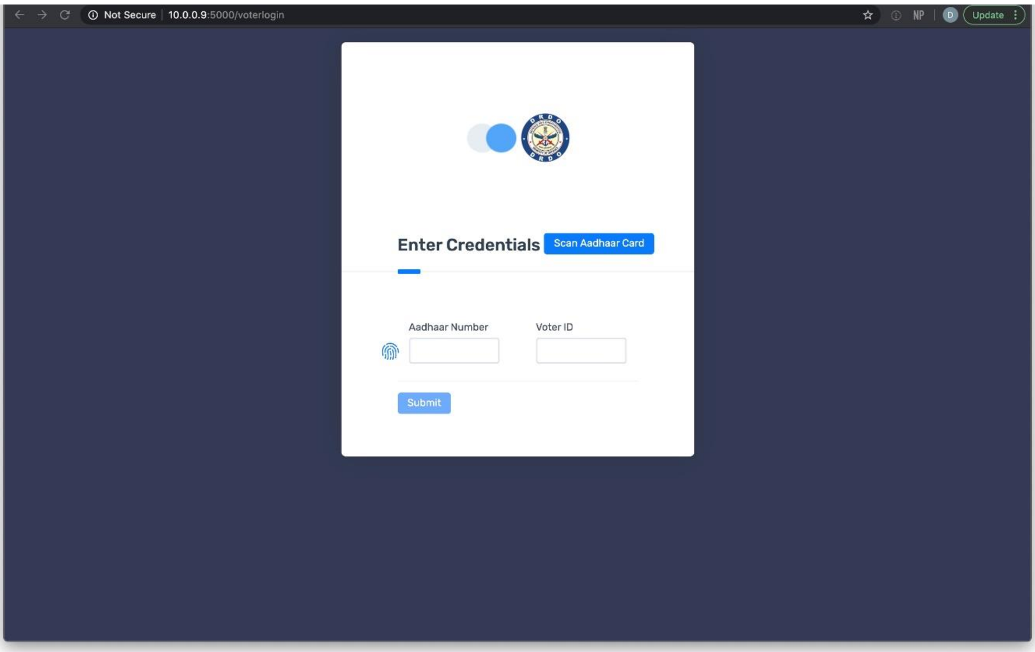
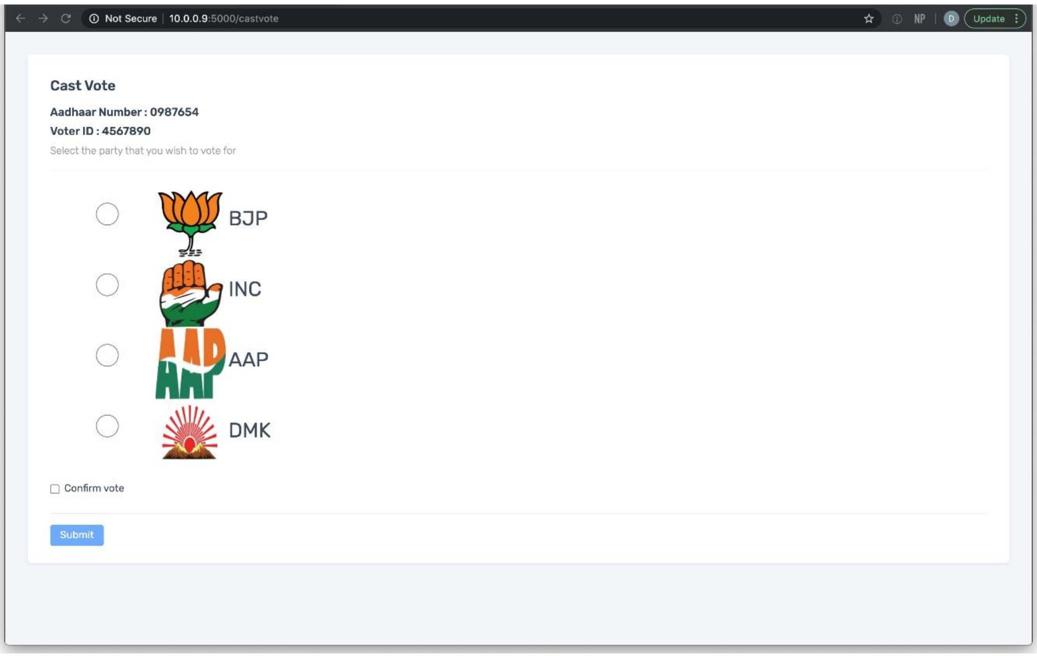


Fig 6.1 Voter logs in here to begin casting vote

Once logged in, the voter will be allowed to pick a party of their choice and confirm their submission.


Not Secure | 10.0.0.9:5000/castvote

Cast Vote

Aadhaar Number : 0987654

Voter ID : 4567890

Select the party that you wish to vote for

○ BJP

○ INC

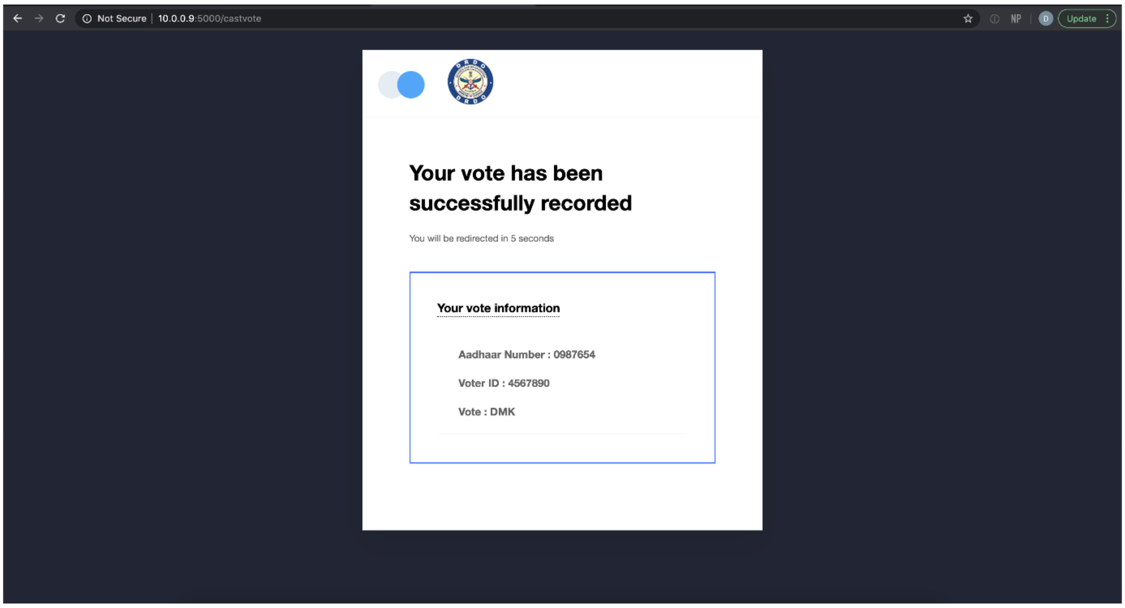
○ AAP

○ DMK

□ Confirm vote

Fig 6.2 Voter casts his vote.

Once vote is cast, he/she clicks on submit. This takes them to the vote confirmation page which displays for a duration of 5 seconds before redirecting to the voter login for the next person to vote.


Not Secure | 10.0.0.9:5000/castvote

Your vote has been successfully recorded

You will be redirected in 5 seconds

Your vote information

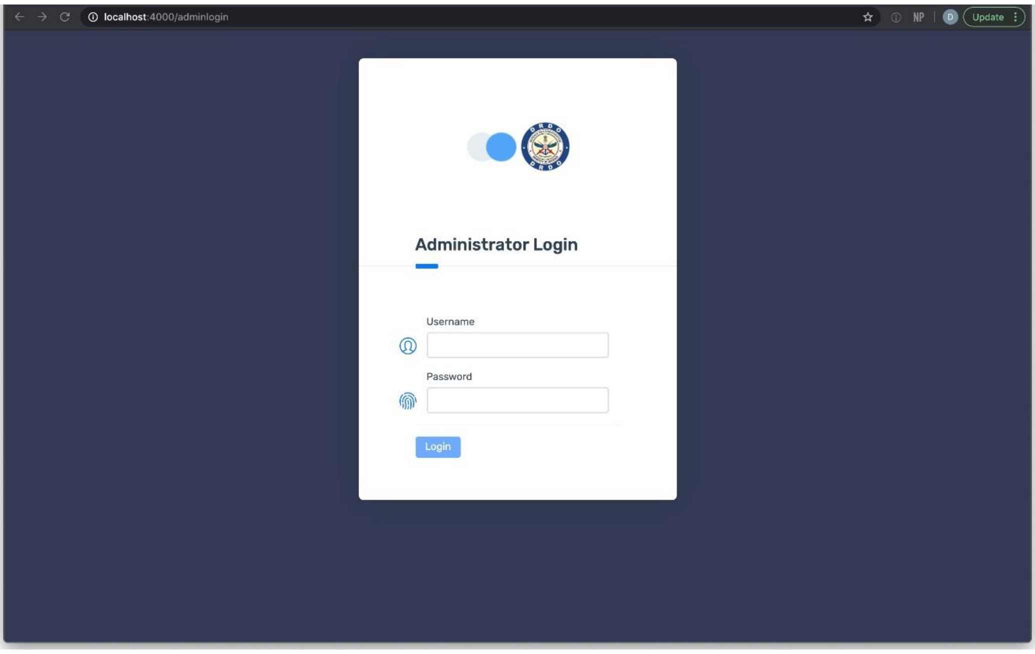
Aadhaar Number : 0987654

Voter ID : 4567890

Vote : DMK

Fig 6.3 Vote confirmation

Once votes are cast, the admin will the able to view the vote division between the different parties. This is done when the admin logs in from the admin login page.



localhost:4000/adminlogin

Administrator Login

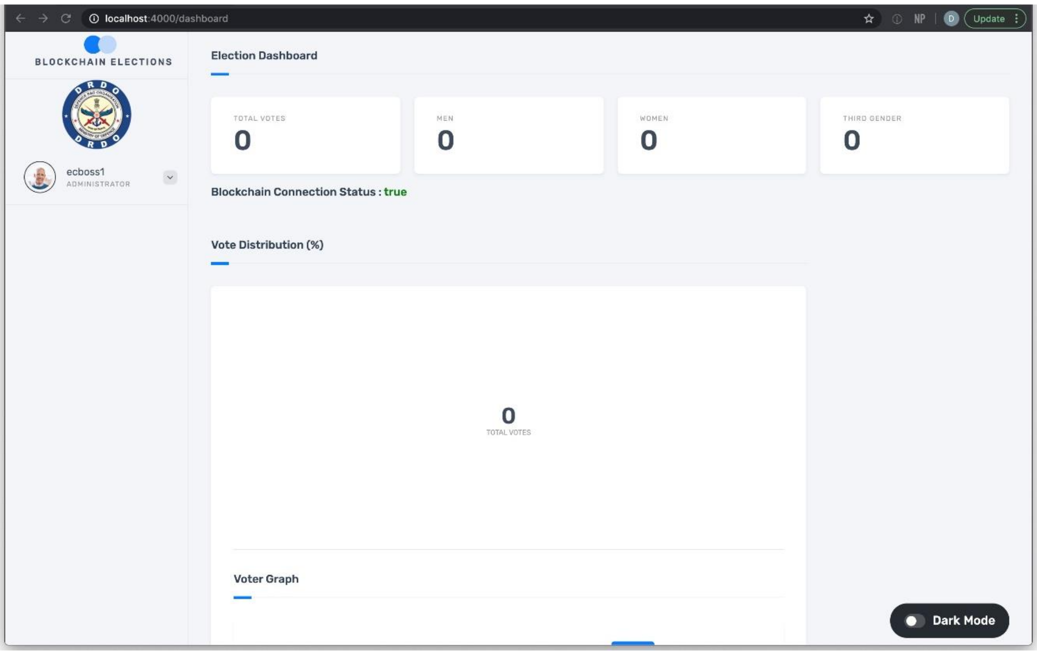
Username

Password

Login

Fig 6.4 Admin login

Once logged in, they can view the admin dashboard with the information about how many votes each party is getting, the percentage of votes for each party and even the male/female/other turnout.



localhost:4000/dashboard

BLOCKCHAIN ELECTIONS

ecboss1
ADMINISTRATOR

Election Dashboard

TOTAL VOTES	MEN	WOMEN	THIRD GENDER
0	0	0	0

Blockchain Connection Status : true

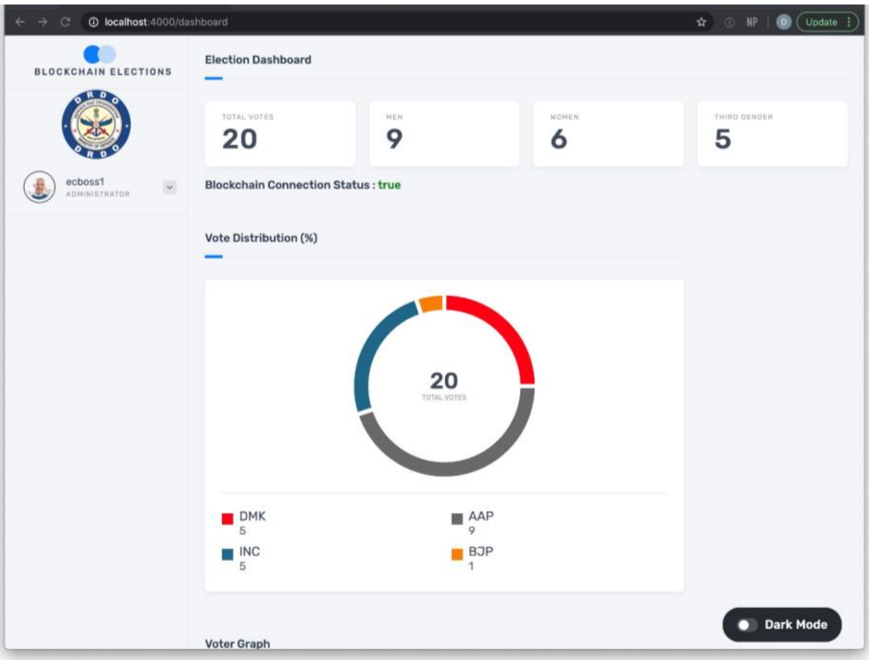
Vote Distribution (%)

0
TOTAL VOTES

Voter Graph

Dark Mode

Fig 6.5 Dashboard before voting



Election Dashboard

TOTAL VOTES	MEN	WOMEN	THIRD GENDER
20	9	6	5

Blockchain Connection Status : true

Vote Distribution (%)

20
TOTAL VOTES

🟥 DMK 5
🟥 AAP 9
🟦 INC 5
🟧 BJP 1

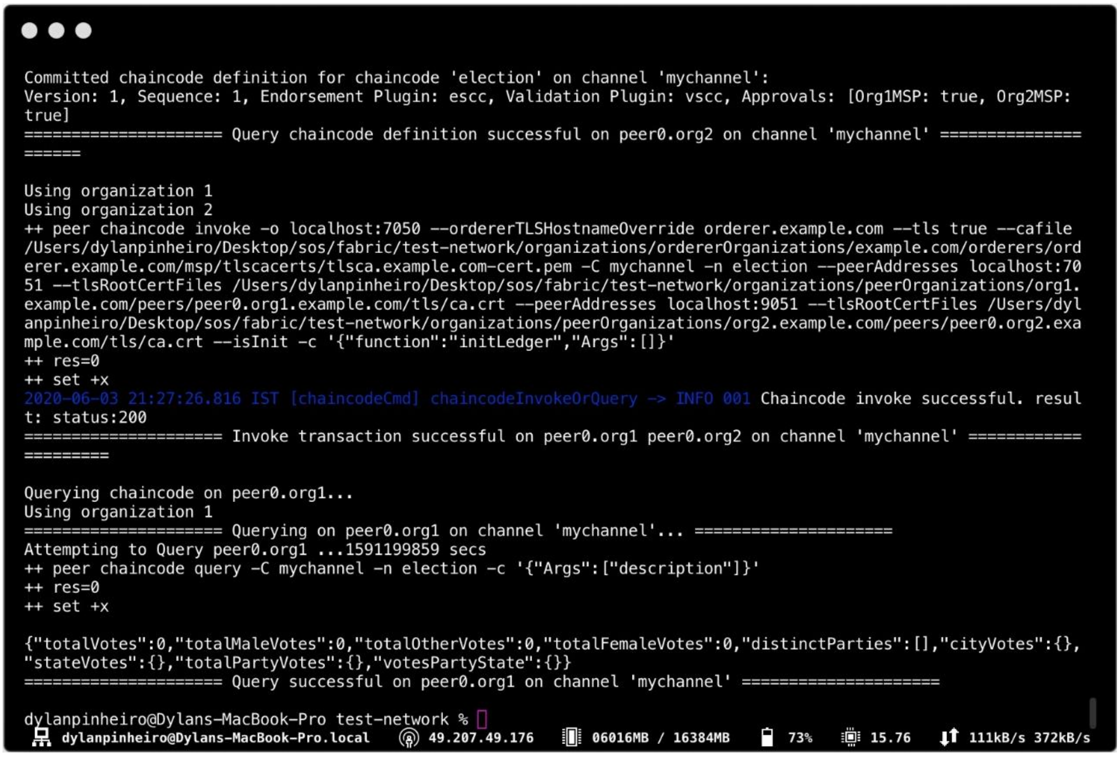
Voter Graph

Fig 6.6 Dashboard after voting

CHAPTER 7

TESTING

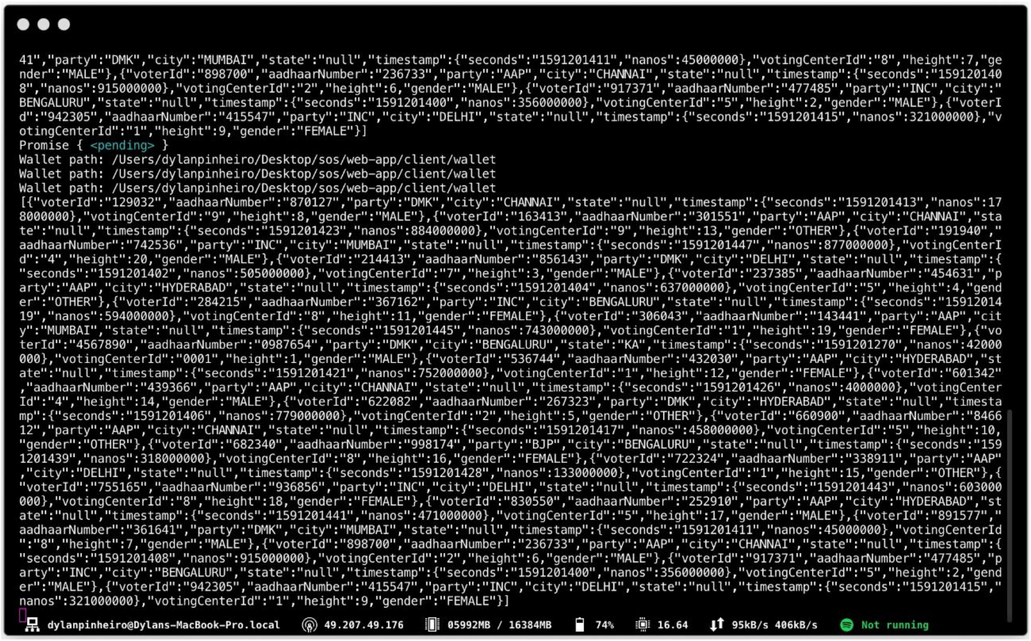
Our main testing was done on the Hyperledger Fabric on which the blockchain runs. Since it is a relatively new technology the documentation is ever changing and produces a lot of bugs with every update. For the purposes of this project, we have gone with Hyperledger Fabric 2.0, the latest release. Figure 7.1 shows the successful invocation of the chaincode. Only once this is done can the blockchain code run.



```
Committed chaincode definition for chaincode 'election' on channel 'mychannel':
Version: 1, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vscc, Approvals: [Org1MSP: true, Org2MSP:
true]
===================== Query chaincode definition successful on peer0.org2 on channel 'mychannel' ===============
======

Using organization 1
Using organization 2
++ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls true --cafile
/Users/dylanpinheiro/Desktop/sos/fabric/test-network/organizations/ordererOrganizations/example.com/orderers/ord
erer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C mychannel -n election --peerAddresses localhost:70
51 --tlsRootCertFiles /Users/dylanpinheiro/Desktop/sos/fabric/test-network/organizations/peerOrganizations/org1.
example.com/peers/peer0.org1.example.com/tls/ca.crt --peerAddresses localhost:9051 --tlsRootCertFiles /Users/dyl
anpinheiro/Desktop/sos/fabric/test-network/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.exa
mple.com/tls/ca.crt --isInit -c '{"function":"initLedger","Args":[]}'
++ res=0
++ set +x
2020-06-03 21:27:26.816 IST [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. resul
t: status:200
===================== Invoke transaction successful on peer0.org1 peer0.org2 on channel 'mychannel' ============
=========

Querying chaincode on peer0.org1...
Using organization 1
===================== Querying on peer0.org1 on channel 'mychannel'... =====================
Attempting to Query peer0.org1 ...1591199859 secs
++ peer chaincode query -C mychannel -n election -c '{"Args":["description"]}'
++ res=0
++ set +x

{"totalVotes":0,"totalMaleVotes":0,"totalOtherVotes":0,"totalFemaleVotes":0,"distinctParties":[],"cityVotes":{},
"stateVotes":{},"totalPartyVotes":{},"votesPartyState":{}}
===================== Query successful on peer0.org1 on channel 'mychannel' =====================

dylanpinheiro@Dylans-MacBook-Pro test-network % 
 dylanpinheiro@Dylans-MacBook-Pro.local   49.207.49.176   06016MB / 16384MB   73%   15.76   111kB/s 372kB/s
```

Fig 7.1 Chaincode invocation

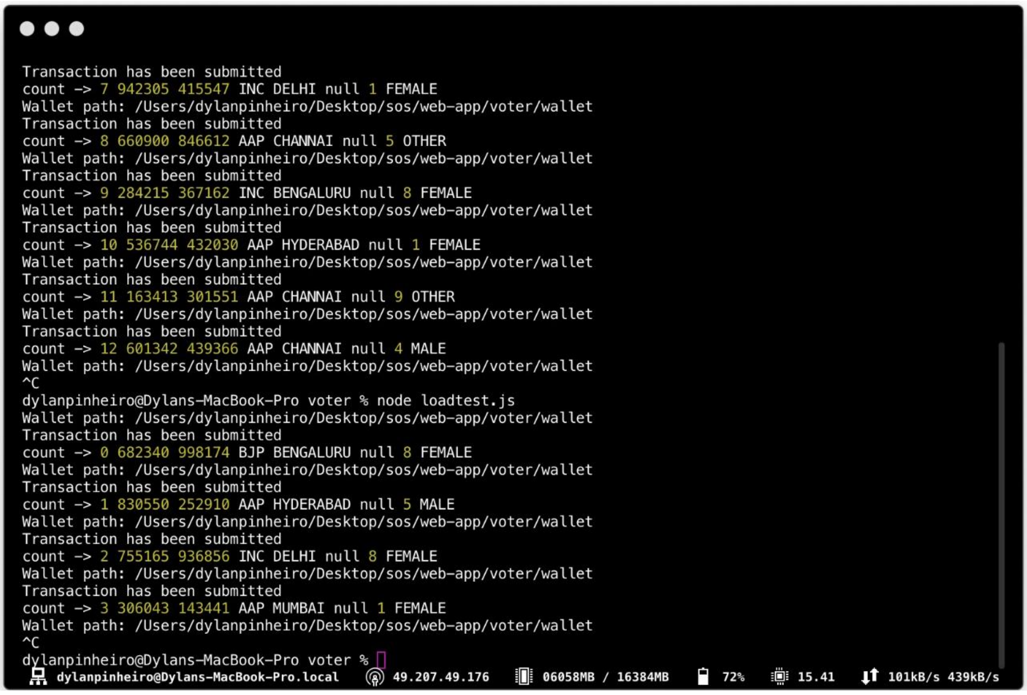
Once this is done, the blockchain code can be run. This ensures that each vote cast keeps getting added to the blockchain as shown in Figure 7.2



```
41","party":"DMK","city":"MUMBAI","state":"null","timestamp":{"seconds":"1591201411","nanos":45000000},"votingCenterId":"8","height":7,"ge
nder":"MALE"},{"voterId":"898700","aadhaarNumber":"236733","party":"AAP","city":"CHANNAI","state":"null","timestamp":{"seconds":"159120140
8","nanos":915000000},"votingCenterId":"2","height":6,"gender":"MALE"},{"voterId":"917371","aadhaarNumber":"477485","party":"INC","city":"
BENGALURU","state":"null","timestamp":{"seconds":"1591201400","nanos":356000000},"votingCenterId":"5","height":2,"gender":"MALE"},{"voterI
d":"942305","aadhaarNumber":"415547","party":"INC","city":"DELHI","state":"null","timestamp":{"seconds":"1591201415","nanos":321000000},"v
otingCenterId":"1","height":9,"gender":"FEMALE"}]
Promise { <pending> }
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/client/wallet
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/client/wallet
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/client/wallet
[{"voterId":"129032","aadhaarNumber":"870127","party":"DMK","city":"CHANNAI","state":"null","timestamp":{"seconds":"1591201413","nanos":17
8000000},"votingCenterId":"9","height":8,"gender":"MALE"},{"voterId":"163413","aadhaarNumber":"301551","party":"AAP","city":"CHANNAI","sta
te":"null","timestamp":{"seconds":"1591201423","nanos":884000000},"votingCenterId":"9","height":13,"gender":"OTHER"},{"voterId":"191940","
aadhaarNumber":"742536","party":"INC","city":"MUMBAI","state":"null","timestamp":{"seconds":"1591201447","nanos":877000000},"votingCenterI
d":"4","height":20,"gender":"MALE"},{"voterId":"214413","aadhaarNumber":"856143","party":"DMK","city":"DELHI","state":"null","timestamp":{
"seconds":"1591201402","nanos":505000000},"votingCenterId":"7","height":3,"gender":"MALE"},{"voterId":"237385","aadhaarNumber":"454631","p
arty":"AAP","city":"HYDERABAD","state":"null","timestamp":{"seconds":"1591201404","nanos":637000000},"votingCenterId":"5","height":4,"gend
er":"OTHER"},{"voterId":"284215","aadhaarNumber":"367162","party":"INC","city":"BENGALURU","state":"null","timestamp":{"seconds":"15912014
19","nanos":594000000},"votingCenterId":"8","height":11,"gender":"FEMALE"},{"voterId":"306043","aadhaarNumber":"143441","party":"AAP","cit
y":"MUMBAI","state":"null","timestamp":{"seconds":"1591201445","nanos":743000000},"votingCenterId":"1","height":19,"gender":"FEMALE"},{"vo
terId":"4567890","aadhaarNumber":"0987654","party":"DMK","city":"BENGALURU","state":"KA","timestamp":{"seconds":"1591201270","nanos":42000
000},"votingCenterId":"0001","height":1,"gender":"MALE"},{"voterId":"536744","aadhaarNumber":"432030","party":"AAP","city":"HYDERABAD","st
ate":"null","timestamp":{"seconds":"1591201421","nanos":752000000},"votingCenterId":"1","height":12,"gender":"FEMALE"},{"voterId":"601342"
,"aadhaarNumber":"439366","party":"AAP","city":"CHANNAI","state":"null","timestamp":{"seconds":"1591201426","nanos":4000000},"votingCenter
Id":"4","height":14,"gender":"MALE"},{"voterId":"622082","aadhaarNumber":"267323","party":"DMK","city":"HYDERABAD","state":"null","timesta
mp":{"seconds":"1591201406","nanos":779000000},"votingCenterId":"2","height":5,"gender":"OTHER"},{"voterId":"660900","aadhaarNumber":"8466
12","party":"AAP","city":"CHANNAI","state":"null","timestamp":{"seconds":"1591201417","nanos":458000000},"votingCenterId":"5","height":10,
"gender":"OTHER"},{"voterId":"682340","aadhaarNumber":"998174","party":"BJP","city":"BENGALURU","state":"null","timestamp":{"seconds":"159
1201439","nanos":318000000},"votingCenterId":"8","height":16,"gender":"FEMALE"},{"voterId":"722324","aadhaarNumber":"338911","party":"AAP"
,"city":"DELHI","state":"null","timestamp":{"seconds":"1591201428","nanos":133000000},"votingCenterId":"1","height":15,"gender":"OTHER"},{
"voterId":"755165","aadhaarNumber":"936856","party":"INC","city":"DELHI","state":"null","timestamp":{"seconds":"1591201443","nanos":603000
000},"votingCenterId":"8","height":18,"gender":"FEMALE"},{"voterId":"830550","aadhaarNumber":"252910","party":"AAP","city":"HYDERABAD","st
ate":"null","timestamp":{"seconds":"1591201441","nanos":471000000},"votingCenterId":"5","height":17,"gender":"MALE"},{"voterId":"891577","
aadhaarNumber":"361641","party":"DMK","city":"MUMBAI","state":"null","timestamp":{"seconds":"1591201411","nanos":45000000},"votingCenterId
":"8","height":7,"gender":"MALE"},{"voterId":"898700","aadhaarNumber":"236733","party":"AAP","city":"CHANNAI","state":"null","timestamp":{
"seconds":"1591201408","nanos":915000000},"votingCenterId":"2","height":6,"gender":"MALE"},{"voterId":"917371","aadhaarNumber":"477485","p
arty":"INC","city":"BENGALURU","state":"null","timestamp":{"seconds":"1591201400","nanos":356000000},"votingCenterId":"5","height":2,"gend
er":"MALE"},{"voterId":"942305","aadhaarNumber":"415547","party":"INC","city":"DELHI","state":"null","timestamp":{"seconds":"1591201415","
nanos":321000000},"votingCenterId":"1","height":9,"gender":"FEMALE"}]
 dylanpinheiro@Dylans-MacBook-Pro.local   49.207.49.176   05992MB / 16384MB   74%   16.64   95kB/s 406kB/s   Not running
```

Fig 7.2 Successful blocks added to the Blockchain

For performance testing purposes, we also have a code that randomly generates votes to be added to the blockchain to test the rate at which each block gets added and how it can handle the load.



```
Transaction has been submitted
count -> 7 942305 415547 INC DELHI null 1 FEMALE
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
Transaction has been submitted
count -> 8 660900 846612 AAP CHANNAI null 5 OTHER
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
Transaction has been submitted
count -> 9 284215 367162 INC BENGALURU null 8 FEMALE
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
Transaction has been submitted
count -> 10 536744 432030 AAP HYDERABAD null 1 FEMALE
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
Transaction has been submitted
count -> 11 163413 301551 AAP CHANNAI null 9 OTHER
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
Transaction has been submitted
count -> 12 601342 439366 AAP CHANNAI null 4 MALE
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
^C
dylanpinheiro@Dylans-MacBook-Pro voter % node loadtest.js
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
Transaction has been submitted
count -> 0 682340 998174 BJP BENGALURU null 8 FEMALE
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
Transaction has been submitted
count -> 1 830550 252910 AAP HYDERABAD null 5 MALE
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
Transaction has been submitted
count -> 2 755165 936856 INC DELHI null 8 FEMALE
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
Transaction has been submitted
count -> 3 306043 143441 AAP MUMBAI null 1 FEMALE
Wallet path: /Users/dylanpinheiro/Desktop/sos/web-app/voter/wallet
^C
dylanpinheiro@Dylans-MacBook-Pro voter %
 dylanpinheiro@Dylans-MacBook-Pro.local   49.207.49.176   06058MB / 16384MB   72%   15.41   101kB/s 439kB/s
```

Fig 7.3 Random votes generated and added to the blockchain

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

8.1 Conclusion

E-voting is a potential solution to the lack of interest in voting amongst the young tech savvy population. For e-voting to become more open, transparent, and independently auditable, a potential solution would be base it on blockchain technology. In this project explores the potential of the blockchain technology and its usefulness in the e-voting scheme. The paper proposes an e-voting scheme, which is then implemented. The implementation and related performance measurements are given in the paper along with the challenges presented by the blockchain platform to develop a complex application like e-voting. The paper highlights some shortcomings and presents two potential paths forward to improve the underlying platform (blockchain technology) to support e-voting and other similar applications. Blockchain technology has a lot of promise; however, in its current state it might not reach its full potential. There needs to be concerted effort in the core blockchain technology research to improve is features and support for complex applications that can execute within the blockchain network. we introduced a blockchain-based electronic voting system that utilizes smart contracts to enable secure and cost-efficient election while guaranteeing voters privacy. We have shown that the blockchain technology offers a new possibility to overcome the limitations and adoption barriers of electronic voting systems which ensures the election security and integrity and lays the ground for transparency. Using an Hyperledger private blockchain, it is possible to send hundreds of transactions per second onto the blockchain, utilizing every aspect of the smart contract to ease the load on the blockchain. For countries of greater size, some additional measures would be needed to support greater throughput of transactions per second.

8.2 Future Scope

Since e-Voting using Blockchain is a very new concept there is a lot of scope for future development. Further implementations can include a custom consensus algorithm. It can also include an Aadhaar QR code scanner for automation. For more protection, a face recognition system could also be implemented to increase the security. Advanced systems may include visual implementation for more statistical data.

REFERENCES

[1] Harsha V. Patil, Kanchan G. Rathi, Malati V.Tribhuwan, “A Study on Decentralized E-Voting System Using Blockchain Technology”, Dept. of Computer Science, D .Y. Patil ACS College, Volume: 05 Issue: 11 | Nov 2018, e-ISSN: 2395-0056

[2] Umut Can Çabuk, Eylül Adıgüzel, Enis Karaarslan, “A Survey on Feasibility and Suitability of Blockchain Techniques for the E-Voting Systems”, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 7, Issue 3, March 2018

[3] Pavel Tarasov and Hitesh Tewari, “THE FUTURE OF E-VOTING”, IADIS International Journal on Computer Science and Information Systems Vol. 12, No. 2, pp. 148-165

[4] Gaby G. Dagher, Praneeth Babu Marella, Matea Milojkovic and Jordan Mohler, “BroncoVote: Secure Voting System using Ethereum’s Blockchain”, 4th International Conference on Information Systems Security and Privacy

[5] Ahmed Ben Ayed, “A CONCEPTUAL SECURE BLOCKCHAIN- BASED ELECTRONIC VOTING SYSTEM”, International Journal of Network Security & Its Applications (IJNSA) Vol.9, No.3, May 2017

[6] F. Þ. Hjálmarsson, G. K. Hreiðarsson, M. Hamdaqa and G. Hjálmtýsson, "Blockchain-Based E-Voting System," 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, 2018, pp. 983-986. doi: 10.1109/CLOUD.2018.00151

[7] B. Shahzad and J. Crowcroft, "Trustworthy Electronic Voting Using Adjusted Blockchain Technology," in IEEE Access, vol. 7, pp. 24477-24488, 2019. doi:10.1109/ACCESS.2019.2895670

[8] N. Kshetri and J. Voas, "Blockchain-Enabled E-Voting," in IEEE Software, vol. 35, no. 4, pp. 95-99, July/August 2018. doi: 10.1109/MS.2018.2801546

[9] A. Singh and K. Chatterjee, "SecEVS: Secure Electronic Voting System Using Blockchain Technology," 2018 International Conference on Computing, Power and Communication Technologies (GUCON), Greater Noida, Uttar Pradesh, India, 2018, pp. 863-867. doi: 10.1109/GUCON.2018.8675008

[10] H. Wu and C. Yang, "A Blockchain-Based Network Security Mechanism for Voting Systems," 2018 1st International Cognitive Cities Conference (IC3), Okinawa, 2018, pp. 227-230. doi: 10.1109/IC3.2018.00-15

[11] https://github.com/hyperledger/fabric, accessed on: 21/01/2020

[12] https://docs.docker.com, accessed on: 15/11/2019

[13]https://hyperledger-fabric.readthedocs.io/en/release-2.0/, accessed on: 11/04/2020