

**Towards partial fulfillment for Undergraduate Degree Level Programme
Bachelor of Technology in Computer Engineering**

A Project Report on:

**“Implementing a Secure Online Voting
System Using Blockchain Technology”**

Prepared by:

Admission No.

Student Name

U16CO110

Aanchal Handa

U16CO086

Shivangi Tanwar

U16CO026

Mehak Wadhwani

U16CO060

Nisha Mansoori

Class : B.TECH. IV (Computer Engineering) 8th Semester

Year : 2019-2020

Guided by : Dr. Bhavesh N. Gohil



**DEPARTMENT OF COMPUTER ENGINEERING
SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECHNOLOGY,
SURAT - 395 007 (GUJARAT, INDIA)**

Contents

Student Declaration	ii
Certificate	iii
Abstract	iv
List of Figures	v
List of Tables	vi
List of Abbreviations	vii
1 Introduction	1
1.1 Application and Motivation	1
1.2 Objective	2
1.3 Organization	2
1.4 Outcome	2
2 Theoretical Background and Literature Survey	3
2.1 Blockchain	3
2.1.1 Characteristics of Blockchain	4
2.1.2 Types of Blockchain	5
2.1.3 Distributed ledgers in Blockchain	6
2.1.4 Cryptography in Blockchain	6
2.1.5 Merkle Tree	7
2.1.6 Consensus Protocol	8
2.2 Hyperledger	8
2.2.1 Hyperledger Fabric	9
2.2.2 Architecture	10
2.2.3 Smart Contract	10
2.2.4 Consensus in Hyperledger	11
2.3 Hyperlegder Composer	12
2.3.1 .BNA Files	12
2.4 Electronic Voting System	13
2.4.1 Key Features of E-Voting System	14

2.4.2	Problems with the proprietary voting system	14
2.4.3	How Blockchain can overcome it?	15
3	Proposed System	17
3.1	Exploring the Existing Voting System	17
3.2	Proposed System	18
3.2.1	Why are we opting Hyperledger over Ethereum?	20
3.3	Modular Design	20
4	Implementation Methodology	22
4.1	Peer Network	22
4.2	Blockchain	26
4.3	REST Server API	29
4.4	Client-Side Application	29
5	Results and Analysis	32
6	Conclusion and Future Work	34
	Bibliography	36
	Acknowledgment	39
	Appendix	40




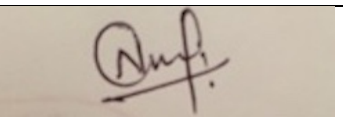
Student Declaration

This is to certify that the work described in this project report has been actually carried out and implemented by our project team consisting of

Sr.	Admission No.	Student Name
1	U16CO110	Aanchal Handa
2	U16CO086	Shivangi Tanwar
3	U16CO026	Mehak Wadhwani
4	U16CO060	Nisha Mansoori

Neither the source code there in, nor the content of the project report have been copied or downloaded from any other source. We understand that our result grades would be revoked if later it is found to be so.

Signature of the Students:

Sr.	Student Name	Signature of the Student
1	Aanchal Handa	
2	Shivangi Tanwar	
3	Mehak Wadhwani	
4	Nisha Mansoori	

Certificate

This is to certify that the project report entitled '*Implementing a Secure Online Voting System Using Blockchain Technology*' is prepared and presented by

Sr.	Admission No.	Student Name
1	U16CO110	Aanchal Handa
2	U16CO086	Shivangi Tanwar
3	U16CO026	Mehak Wadhwani
4	U16CO060	Nisha Mansoori

Final Year of Computer Engineering and their work is satisfactory.

SIGNATURE:

GUIDE

Dr. Bhavesh N. Gohil

JURY

HEAD OF DEPT.

Dr. M A Zaveri

Abstract

Elections are an integral part of any modern democracy, yet still predominant sections of the society fail to cast their vote due to certain flaws in the election systems. Our college elections, though on a smaller scale, still face the same issues. Problems like flawed elections, election booth rigging, tampering with votes and polling booth capture are some of the leading causes of a failing and pointless election. In this project, we discuss how the utilization and incorporation of blockchain technology prohibits and eliminates a large number of problems that are seen in the modern day election process. Blockchain is a distributed ledger technology. Transactions are verified and appended to the blockchain with the help of peers present in the decentralised network. This leads to the creation of an immutable chain of registered events, whose authenticity and veracity can be proved by a consensus protocol. The deployment of chaincodes into the blockchain technology is the cardinal reason why it is possible to integrate this technology with the voting system. The objective of this project is to propose and build an E-voting model that can be applicable for SVNIT college elections, using blockchain as service to implement distributed electronic voting system and make it more efficient.

Keywords: Blockchain E-voting - Distributed ledger - Chaincode - Hyperledger Fabric - Peers
- Composer - REST API

List of Figures

2.1	Conversion of plain text in Hash using SHA-256	7
2.2	New Block created with previous hash and data	7
2.3	Hyperledger Fabric CA structure[25]	9
2.4	Multi-peer Network Architecture	10
2.5	Sequence Diagram: Consensus in Hyperledger Fabric [4]	11
2.6	Application Workflow [23]	12
2.7	Components that reside within a .bna file [30]	13
3.1	Proposed system flowchart	18
3.2	Transaction Flow [29]	19
4.1	Gantt Chart	22
4.2	Docker-compose.yml file	23
4.3	CA file for peer authentication	23
4.4	Server Started	24
4.5	Dockercompose.yml file for peer1	25
4.6	Docker containers status in peer1	25
4.7	Roles of peer machines	25
4.8	Docker ps status	26
4.9	Model file	26
4.10	Logic.js file	27
4.11	Permissions.acl file	27
4.12	The cards used in the system	28
4.13	Composer REST server explorer	29
4.14	Details verification portal	30
4.15	Voting Page	30
4.16	Backend once vote is cast for a sample transaction	31
4.17	Results representation	31

List of Tables

2.1	Block Structure	3
-----	---------------------------	---

List of Abbreviations

1. **EVM** Ethereum Virtual Machine
2. **E-voting** Electronic Voting
3. **PoW** Proof of Work
4. **PoC** Proof Of Concept
5. **P2P** Peer to peer
6. **PoS** Proof Of Stake
7. **SHA** Secured Hash Algorithm
8. **LLL** Lisp Like Language
9. **DFD** Data Flow Diagram
10. **OTP** One time Password
11. **IFF** If and only if
12. **CA** Certificate Authority
13. **SDK** Software Development Kit
14. **API** Application Programming Interface
15. **MIS** Management information system
16. **YAML** YAML Ain't Markup Language
17. **IP** Internet Protocol
18. **JWT** JSON Web Token
19. **SVNIT** Sardar Vallabhbhai National Institute of Technology

Chapter 1

Introduction

1.1 Application and Motivation

A fundamental pillar in any democratic system is the elections, a process that enables the general public to present their views via the use of a vote. The election process as a whole should be reliable and transparent, thus corroborating participants of its reliability and trustworthiness. This is primarily because of the large role they play in today's society. Looking through this perspective, the approach to voting has been a never-ending and ever-evolving realm. The main motivation behind the constant evolution has been to make the system transparent, secure, and verifiable. Since its impact is so vast, efforts have been made to boost the overall efficiency and resilience of a voting system. The creation of E-voting or Electronic voting has certainly made its mark. Since its first use as punched-card ballots in the 1960s, e-voting systems have achieved remarkable progress with its adoption using internet technologies [1]. There are some points that must be adhered to that specify a benchmark parameter so as to ease its widespread adoption. These consist of anonymity of the voter, non-repudiation among others, and integrity of the vote.

Some countries have taken serious measures to improve their voting system, by incorporating decentralized peer to peer networks via the use of blockchain technology. This will be accompanied by a public ledger. The first country to use blockchain technology to verify votes was Sierra Leone. This was done during their nationwide presidential elections in the month of March 2018. The key attribute of blockchain technology is the inability to delete or change information from the pre-existing blocks. The backbone of the blockchain technology is the distributed network built up of a large number of interconnected nodes. Every node in the distributed system has its own copy of the distributed ledger that holds the full history of all the transactions the network has processed. In this kind of network, no single entity or individual has the authority to control the flow. If the majority of the nodes agree, the transaction will be accepted. The anonymity of the user is maintained in this network. An overly simplified statement that allows one to understand blockchain technology suggests that it is an extremely viable basis for electronic voting. In addition, it has the potential to make electronic voting a more acceptable and reliable portal.

1.2 Objective

The main objective is to achieve a decentralized, transparent, and secure e-voting system that leads to greater participation in a more flexible and efficient manner. We would no longer be reliant on staff to provide us with a location, time, and access rights for casting a vote.

1.3 Organization

The upcoming chapters of this report contains theoretical background on the concept of blockchain, and e-voting systems (chapter 2.), proposed system (chapter 3.), an implementation methodology (chapter 4.), results and analysis section (chapter 5.) and Conclusion (chapter 6.).

1.4 Outcome

This project evaluates the use of blockchain as a service to implement an electronic voting (e-voting) system for elections that take place annually during the academic year. This project makes the following original contributions:

1. research existing blockchain frameworks suited for constructing blockchain based e-voting system,
2. propose a blockchain-based e-voting system for college level elections that uses private blockchain to enable liquid democracy

Chapter 2

Theoretical Background and Literature Survey

2.1 Blockchain

Blockchain, invented in the year 2008 was designed to serve as a public transaction ledger of bitcoin, a cryptocurrency. At that time it was under the pseudonym, Satoshi Nakamoto. Blockchain is a distributed database that keeps a chronologically-growing list (chain) of records (blocks) secure from tampering and revision[9]. Each block present in the list contains the hash of its previous block, a timestamp, and previous transaction data. The hash is stored and generated cryptographically. The transaction data is portrayed using a Merkle tree. It is an open distributed ledger that can record transactions between two parties efficiently and in a verifiable and immutable way. To sum it up, a blockchain is a decentralized distributed, and at times public digital ledger that is used to keep track of transactions across a very large network, so that any of the involved parties data cannot be altered without the alteration of all subsequent blocks. Due to this capability of blockchain technology, members are able to audit and verify transactions independently. The entire blockchain database is autonomously managed with the help of the peer to peer network and a distributed time-stamping server.

The blocks that make up the blockchain are built up of digital pieces of information. Generic quantitative information of primary concern are mentioned below in the table:

Field	Description	Size
Block Size	The size of the whole block	400 bytes
Block Header	Encrypted almost unique hash	80 bytes
Transaction Counter	The number of transaction that follow	1 to 9 bytes
Transaction	Contains the transaction saved in the block	Depends on the transaction size

Table 2.1: Block Structure

2.1.1 Characteristics of Blockchain

Decentralization

The decentralized nature of this technology provides a mechanism through which one does not need to rely on a central point of control. Due to the lack of a single authority, it allows for a more fair and considerably more secure system. Blockchain exploits the consensus protocol across a network of nodes, to validate transactions and record data in a manner that is incorruptible. The fees that are normally collected by these central authority organizations are no longer a factor. Thus transacting on the blockchain reduces the cost incurred by the third party wanting to perform the transaction.

Privacy

Three basic and important capabilities that are supported by the blockchain technology implemented in Applications are: (1) the hash chained storage, (2) digital signature, and (3) the commitment consensus for adding a new block to the globally chained storage [28]. In regard to this project, the report will focus on hashing and certification forming. The privacy of each voter is maintained by using hash function(SHA-256) to create the voter id so that after the voting process the votes are anonymous hence the privacy is maintained.

Hashing A hashing algorithm refers to a mathematical function that takes a variable size string as input and transforms(hashes) it into a fixed size string, which is called the hash value, as output[12]. A key property that must be present in this function is that the transformation should be one way and Collision Resistance.

Peer to peer network: Blockchain functions in a manner where nodes in a blockchain system are able to confirm the validity of a transaction rather than using a central authority. Before any block can be appended to the existing chain, nodes must ensure that the transaction is valid. Consensus protocols are called upon such as proof of work and proof of stake deployed by miners for validation of the transactions.

Communication takes place directly between peers, instead of via a central node. Information is spread through the entire network because every node has information stored on what is happening and passes it along to the adjacent nodes. A transaction has to be proposed and then endorsed(by endorsing peers) before it can be submitted successfully to the blockchain. Once submitted, the transaction will make its way to respective peers in the network that are supposed to get this transaction. These peers will then commit and save the transaction into their ledger. Thus these peers will now be denoted as committer peers.

Transparency

Although transparency is maintained, blockchain keeps in pseudonymity in mind. Anyone inspecting the Blockchain is capable of seeing every transaction and its hash value. Someone using

the Blockchain is able to be anonymous if they wish or they can give their identification to others. All that you see on the Blockchain is a record of transactions between Blockchain addresses.

Immutability

The ability for a blockchain ledger to remain a permanent, indelible, and unalterable history of transactions is a definitive feature that blockchain evangelists highlight as a key benefit. This creates trust in the transaction record.[16] Immutability has the potential to transform the auditing process into a quick, efficient, and cost-effective procedure, and bring more trust and integrity to the data businesses use and share every day.

2.1.2 Types of Blockchain

There are three types of Blockchains that have emerged after Bitcoin introduced Blockchain to the world.

1. Public Blockchain

A public blockchain as its name suggests is the blockchain of the public, meaning a kind of blockchain which is *for the people, by the people and of the people*. In public blockchains, no single entity is in charge, and anyone can have the authority to write/read/audit the blockchain. Thus making them blockchains that are transparent and open. This means that anyone at any given point can review any piece of information on any relevant block.

Ethereum is an open software platform based on public blockchain technology that enables developers to build and deploy decentralized applications [13]. In this form of distributed blockchain network, Ethereum relies on miners to earn Ether. Ether is a form of crypto token that propagates the running of the network. Apart from the fact that it is a tradable cryptocurrency, Ether is also used to pay for transaction fees and developers use it to pay the miners for their services on the Ethereum network. An Ethereum Virtual Machine is used to solve complex and high degree computational problems. EVMs are the base of any system of blockchain. When working alongside the Ether, they together form the powerhouse of smart contracts.

Various decentralized consensus mechanisms are the means through which decision making takes place on the public blockchain. A few such mechanisms are Proof of Work (POW) and Proof of Stake (POS) etc. The original consensus algorithm present in any blockchain network is the Proof of Work Protocol. On the basis of this protocol, transactions are confirmed and a new block is appended to the chain. Miners are working and competing against each other to gain their reward, Ethers. Users send each other digital tokens in the blockchain network. A ledger keeps a record of all the transactions into the block. A lot of care must be taken whilst confirming a transaction, and arrange blocks. This hefty load is placed on the miners, via the process of mining.

Example: Bitcoin, Litecoin

2. Private Blockchain

In the case of a private blockchain, it is under the ownership of an individual or an organization. This type of blockchain differs from the public blockchain as there is an entity who is in charge of important things such as the read and write operations. They can decide who to selectively give access to read or vice versa. The consensus mechanism exists in this type of blockchain as well. It is achieved with the help of the central members who hold the ability to give mining rights to whichever peer member in the network. This makes this more skewed towards the centralized system again, where the various rights have been only provided to the central trusted authority. Yet it is cryptographically secured from the firm's perspective and is an extremely cost-effective strategy for them.

Example: Bank Chain

3. Consortium or Federated Blockchain

A federated blockchain tries to remove the presence of the central trusted member present in a private blockchain. Instead of having just one entity in charge, Consortium states it is advantageous to have a group of representatives that will come together and make decisions that are in the best interest of the entire network. This group of individuals is referred to as a federation or a consortium, hence the name has been derived from this.

Example: r3, EWF.

2.1.3 Distributed ledgers in Blockchain

A database that by nature is consensually shared and has the ability to be synchronized across various peers, sites, or geographies, is known as a distributed ledger. Whenever a transaction takes place, there is the creation of public witnesses", thereby making a cyberattack more difficult. Every participant in the network can access the recordings shared and have the ability to copy it to their own hash value. All of the information on a block is accurately and securely saved using cryptography and one has the power to access this information using cryptographic signatures and keys.

It becomes an immutable database, once this information is saved cryptographically to the block. On the contrary, centralized ledgers are prone to cyber-attacks, distributed ledgers have held their place as they are inherently harder to attack. This is primarily due to the fact that all the distributed copies need to be attacked at the same time for an attack to have done its malicious deed. Another plus point that must be brought to light is that these records are resistant to changes by a single party.

2.1.4 Cryptography in Blockchain

Hashes

Hashes are used to represent the current state of a system and in this case the state of a blockchain. The input represents everything that has happened on a blockchain, so every single transaction up to that point, combined with the new data that is being added. What this means is that the output is based on, and therefore shaped by, all previous transactions that have occurred on a blockchain.

A genesis block, as referred to as the first block in the blockchain network, is made up of

its transaction data. This when merged together with details like when the transaction was done, creates a unique hash, which is validated. This hash value along with the upcoming new transaction will be merged together cryptographically to create a new hash that will then be used with the following transaction. This cycle repeats itself for every new addition to the blockchain. This means that every node is linked back to its previous node because of its hash function. This forms a chain back to the first node in the transaction, the genesis block. With the means of this mechanism, a transaction can be added securely until and unless the nodes on the network are all on the same page and following the same consensus on what the hash value should be.

The hash value for each block is placed in the header which allows anyone to identify the block. To create the hash value blockchain technology makes use of the Secure Hash Algorithm (SHA-256), a one-way function, which gives an almost idiosyncratic fixed 256-bit hash value. This particular algorithm can take any size plain text as input and encrypt it to a 256-byte binary value.

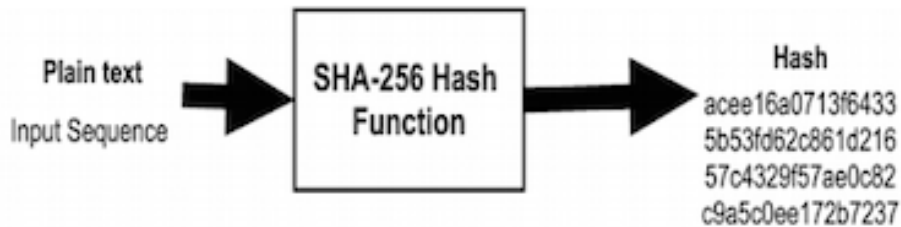


Figure 2.1: Conversion of plain text in Hash using SHA-256

Each header contains information that links a block to its previous block in the chain. The primary identifier of each block is the encrypted hash present in the header. A digital fingerprint is created by combining the two informations present: the information concerning the new voter, and the previous hash value.

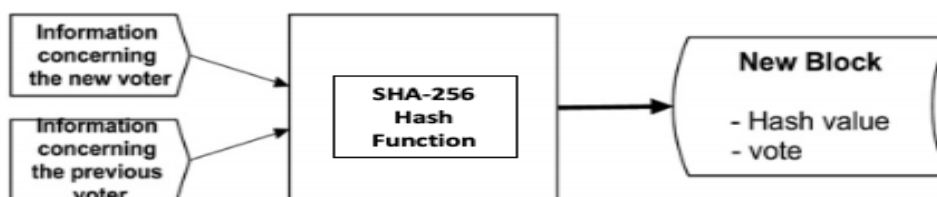


Figure 2.2: New Block created with previous hash and data

2.1.5 Merkle Tree

It is a fundamental concept of Blockchains. A Merkle tree summarizes all transactions in a block by producing a digital fingerprint of the entire set of transactions. This allows the user to verify and vouch for the presence of a transaction in a block. A Merkle tree is created by repeatedly hashing

pairs of nodes until there is only one hash left. This remaining hash is referred to as the root hash. This tree is constructed in a bottom-up manner, from individual hashes of each transaction. Each non leaf node is a hash of the previous and a leaf node represents a hash of transactional data.

The Merkle root summarizes all of the data in the related transactions and stores it in the block header. This allows for data integrity as if even a single detail in any transaction is changed, the Merkle root also changes.

2.1.6 Consensus Protocol

The consensus protocol tells the network which of the peer nodes provide a guaranteed ordering of the transactions. Apart from this, a consensus protocol also validates the block of transactions that needs to be added to the ledger. A consensus must adhere to the following in the network:

- Creates an interface, and a dependency on the smart contracts layer to verify the correctness of an ordered set of transactions in a block.
- Guarantees the accuracy of all transactions proposed in the block, sticking to the consensus policies and endorsements.
- Must reach a mutual agreement over the order, correctness, and indirectly the result of execution.

Consensus properties

Consensus must satisfy two properties to guarantee agreement among nodes: safety and liveness.

1. Safety means that each node is guaranteed the same sequence of inputs and results in the same output on each node. When the nodes receive an identical series of transactions, the same state changes will occur on each node. The algorithm must behave identically to a single node system that executes each transaction atomically one at a time. [18]
2. Liveness means that each non-faulty node will eventually receive every submitted transaction, assuming that communication does not fail.

2.2 Hyperledger

Hyperledger is a technology that is created with the motivation to promote an open-source collaborative effort in order to allow the blockchain technology in various other verticals. Various leaders of finance, banking, supply chain, and technology have come together in collaboration. This technology does not support bitcoin. Its main focus is on the generation of new transactional applications that establish transparency, accountability, and trust. This must happen alongside boosting the efficiency of business processes and strategies.

2.2.1 Hyperledger Fabric

Hyperledger Fabric is an enterprise-grade permissioned distributed ledger framework for developing solutions and applications [15]. A broad spectrum of industries utilizes their versatile and modular design. The Hyperledger fabric enables performance without hindering the privacy constraints, via a consensus. Within private industrial networks, the verifiable identity of a participant is a primary requirement[17]. All network participants should have a known identity in the case where permissible membership exists. This is the case in Hyperledger fabrics.

Hyperledger Fabric has a modular layout, which is segregated into three different stages. The first stage is the smart contracts that make up the distributed logic processing. This then transitions to the next stage where transaction ordering takes place. The final stage is the validation and committing of the transactions into the blockchain. There are multiple benefits of the segregation, such as improving the network scalability, decreasing the number of trust levels and verification that keeps the network running smoothing and in turn yields a better overall performance.

Hyperledger Fabric CA

The Hyperledger Fabric CA is a Certificate Authority (CA) for Hyperledger Fabric. Various features such as assurance of enrollment certificates (ECerts), registration of identities and certificate revocation, and renewal are available through the Hyperledger Fabric CA.

There are only two ways of interacting with the Fabric CA server. The primary source is through the Fabric CA client or via the Fabric SDKs. The entire communication process is conducted with the help of REST APIs. Applications interact with the network with the help of APIs. These in turn run smart contracts. These smart contracts or chain codes are hosted by the network and have a unique name and version associated with them. APIs are accessible with a software development kit, or SDK.

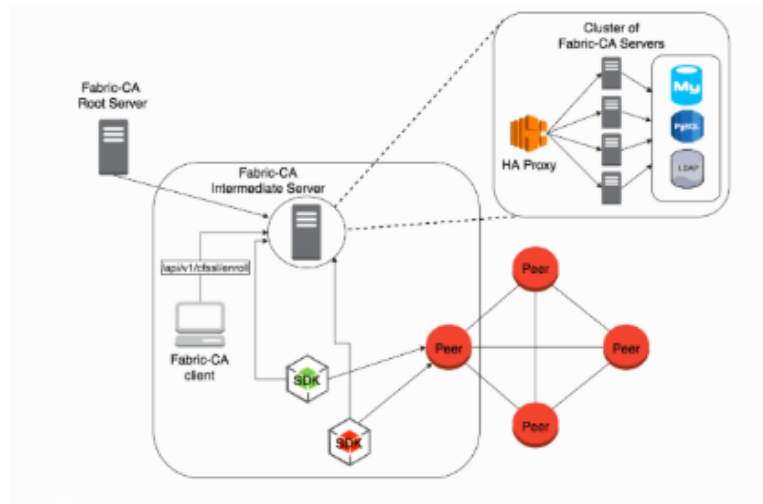


Figure 2.3: Hyperledger Fabric CA structure[25]

2.2.2 Architecture

Hyperledger Fabrics has its own set of components that work together in harmony. Each piece is integral to the working of the system and has its own functions to play. Every component runs on its own Docker instance and is engineering to work side by side with each other. The docker instances can communicate with each other regardless of the fact that they are running on various physical machines on the network.

Certificate Authority. All the access control logic, permissions for the users and managing the identities of every member in the hyperledger blockchain network is the responsibility of the certificate authority as explained in the above section.

Orderer Keeping every component and member functioning in a synchronized state is the main function of the orderer. The orderer is the one informing all the peers present in the system about any new transactions that have been committed. The more the number of orderers the less the number of faults the system faces.

Peers Every peer in the network has its own copy of the world state, and only peers have the permission to commit transactions in the network. The information is also passed onto CouchDB, which acts like the database. Every system can have multiple peers, and anchor peers. Anchor peers are pivotal to communicate with other organizations.

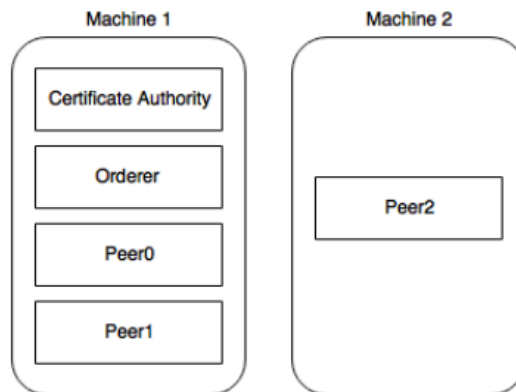


Figure 2.4: Multi-peer Network Architecture

2.2.3 Smart Contract

A computer program stored on the blockchain network itself is what is referred to as smart contract. Its basic function is to execute the terms defined and laid out in the program. When invoked by an external event, or if some preset condition has been fulfilled, then only does the smart contract begin executing. Contract-oriented high-level languages such as LLL, Serpent or Solidity are used in EVMs.

2.2.4 Consensus in Hyperledger

Hyperledger makes use of the permissioned voting-based consensus from the pool of other consensus named the lottery-based consensus. The operating assumption for Hyperledger developers is that business blockchain networks will operate in an environment of partial trust [18]. To sum up, the voting-based algorithm is more advantageous. This is primarily due to the fact that it offers a low latency method. When the major proportion of the nodes validate a block to enter the chain, consensus prevails and finality occurs. There is a massive tradeoff in the consensus in Hyperledger. The voting-based algorithm is in need of nodes that transfer information to all the other nodes existing in the network, yet the more nodes that are present in the network, one sees a longer time is required to reach a consensus. Thus there exists the tradeoff, between time and scalability.

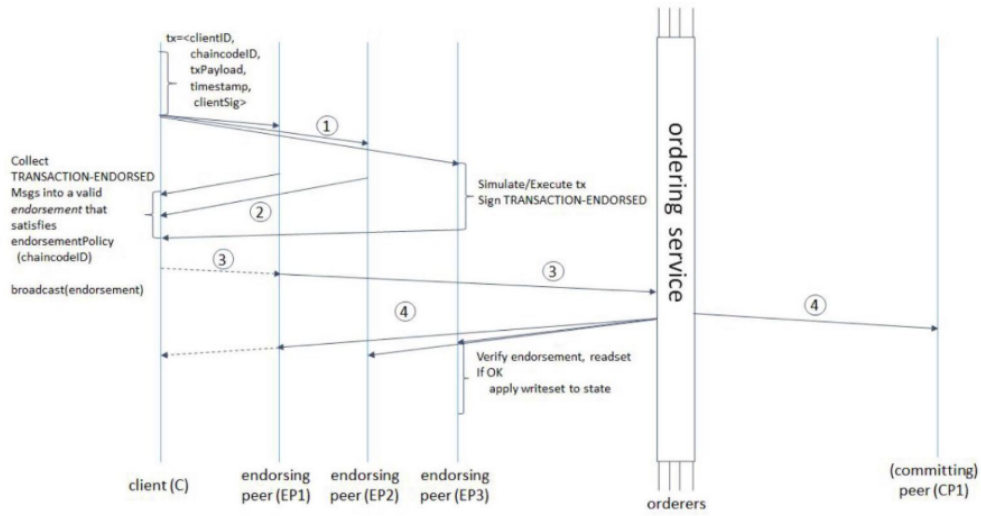


Figure 2.5: Sequence Diagram: Consensus in Hyperledger Fabric [4]

2.3 Hyperledger Composer

A Hyperledger Composer is a set of tools that aid in the creation of the blockchain network. It is implemented by various business owners, and developers to blockchain based applications and smart contracts. The primary aim behind this is to help solve business problems and/or to boast operational efficiency. It has an inbuilt function that provides the user with a DSL to state transactions, assets and participants in the network.

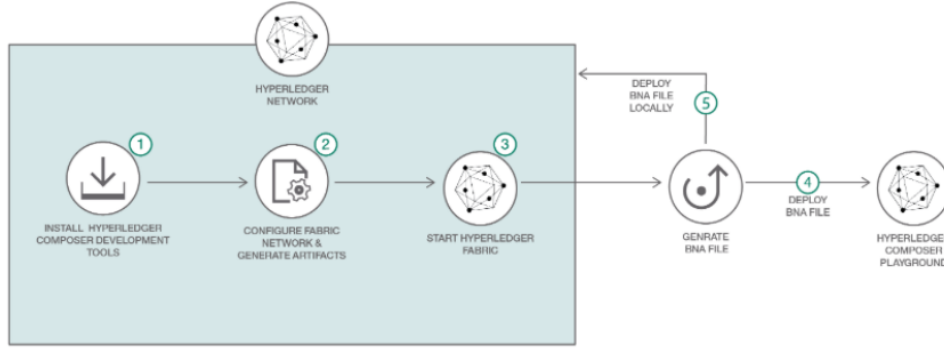


Figure 2.6: Application Workflow [23]

2.3.1 .BNA Files

When trying to build a business network, access controls must be kept in mind. They are enforced as per the access control rules laid out in the Business Network definition. Before the business network definition can be deployed, it must be compressed and packed into a Business Network Archive (.bna) file. Each business network must have at least one participant, and that participant must have a valid identity for accessing the business network [19]. Without this, client applications will be unable to interact with the business network. A business network administrator is a participant who is responsible for configuring the business network for their organisation after the business network is deployed, and is responsible for on-boarding other participants from their organisation [19].

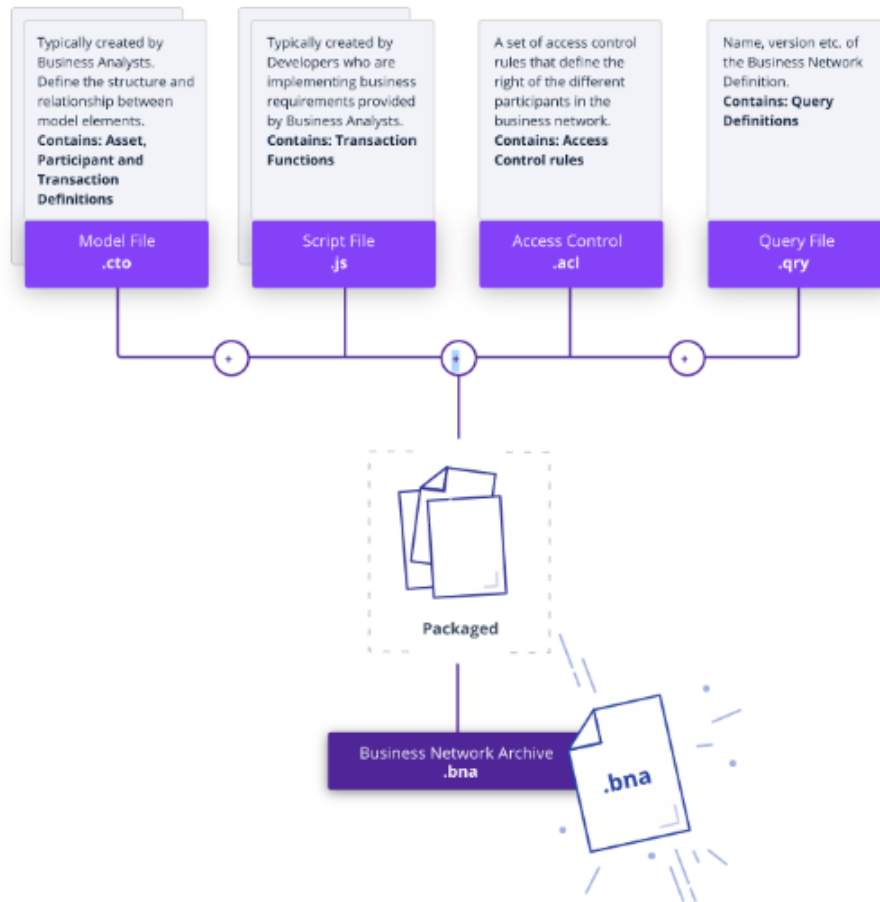


Figure 2.7: Components that reside within a .bna file [30]

2.4 Electronic Voting System

In the early 80s, David Shaum brought the concept of electronic voting to light. The system made use of public-key cryptography. This helped cast votes and keep voter identities hidden. The Blind Signature Theorem was used in order to have no links between voter and ballots.

For the past several years, many governments have taken a keen liking towards e-voting systems. This interest, however, has been followed by security issues that start to become prominent upon usage. There have been quite a few methods proposed which result in a more transparent voting system, but in the end they are both costly and arent feasible to be implemented on a large scale. As the population becomes more tech-savvy, electronic and remote voting becomes an incentive for greater participation in democracy.

Replacing the archaic traditional schemes used to cast votes, a newer, more modern take via the electronic voting system must keep in mind various points. These include the detection of fraud, allowing the voting process to be verifiable and also traceable. In this work, we discuss the desired criteria of electronic voting, and how beneficial blockchain technology is in this vertical. A

Transparent, cost-effective method to verify the transactions that take place in the large scale voting system are some of the reasons behind promoting the use of blockchain.

2.4.1 Key Features of E-Voting System

All together for an e-voting ballot system to be regarded secure certain formally-expressed properties must hold true.

1. Fairness

No early results should be obtainable before the finish of the casting a ballot procedure; this gives the affirmation that the rest of the voters won't be influenced in their vote.

2. Eligibility

This property expresses that only eligible voters should be permitted to make their vote and they should do so only once. The premise of this property is verification since voters need to demonstrate their identity before being considered eligible or not.

3. Privacy

This property expresses that only eligible voters should be permitted to make their vote and they should do so only once. The premise of this property is verification since voters need to demonstrate their identity before being considered eligible or not.

4. Verifiable

This property guarantees that all gatherings included can check whether their votes have been counted or not. Normally two types of verifiability are defined, individual and all-inclusive verifiability. Individual verifiability enables an individual voter to check that one's vote has been counted. All-inclusive verifiability necessitates that anybody can check that the election result is the one distributed.

5. Coercion-Resistant

A coarser should not be able to recognize whether a constrained voter cast a ballot the manner in which they were told to. It is inside the extent of the paper to consider a convention that has the previously mentioned properties. Be that as it may, Coercion resistance won't be effectively sought after since it was regarded unrealistic to be accomplished simply with mechanical methods in a remote e-vote a ballot convention. The convention does anyway have the property of Forgiveness that can be seen as a flimsier thought of the compulsion opposition property. A coarser should not be able to tell whether a constrained voter has cast their ballot under pressure. Coercion resistance won't be effectively sought after since it was regarded unrealistic to be accomplished simply with mechanical methods in a remote e-vote a ballot convention[2].

2.4.2 Problems with the proprietary voting system

The existing e-voting system faces serious design flaws. They are proprietary in that there is a single supply that controls the code base, database, and the system outputs and supplies the monitoring tools at the same time. In other words, the design is centralized. The lack of an open-source, independently verifiable output makes it difficult for such centralized systems to acquire

the trustworthiness required by voters and election organizers. Without an easy, free access to effective, secure electronic voting technology, political participation is limited to on-site elections that are few and far between, given the high setup cost of election preparation, supervision, and post-election operations. This design flaw is therefore barricading electronic vote applications at a time when growing technology literacy and usage is present. This should in fact foster their widespread adoption of utilizing blockchain in the voting industry.

2.4.3 How Blockchain can overcome it?

If the blockchain method is applied to the online voting system, it will be possible to implement electronic management. It can also be used to check whether voters voted or count votes, making the voting system more efficient.

The advantages of the blockchain-based online voting system are as follows: -

1. Time and cost reduction

The blockchain-based online voting system can reduce time and cost compared to conventional voting. According to the existing voting method, it takes some time before the ballot counter counts all the votes, but if the voting is done on the blockchain server, it will be possible to see the voting result immediately after voting. So it is not necessary to wait for the voting result to come in after a certain time. Also, due to the simplified voting process, voting costs can also be expected to be reduced. The blockchain-based online voting system can reduce the cost of voting and ballot counting compared to that of the conventional voting method, and unlike the existing online voting system, the blockchain-based online voting system can also reduce the cost of implementing the central server and the security system.

2. Increased participation of voters

The blockchain-based online voting system can increase citizens participation in voting. Everyone finds it difficult to participate in the current direct voting, but if the blockchain is used for voting, it will be possible to overcome physical limitations, and more people can participate in the policy-making process. Also, as the online ballot paper provided a link to the information on each candidate, they could get information more easily and faster than in the conventional voting.

3. Security and reliability

With regard to the online voting system, there are concerns about the secrecy of voting, the personal information security issue and the abuse and fabrication of the right to vote. However, the decentralized information sharing system of the blockchain can secure integrity and security on its own. In the conventional online voting system, as the central server and the central database manage and process voting values, there is a high risk of fabricating voting results.

In the blockchain-based online voting system, not all voting values are stored in the central server and the central database, but they are disclosed to everyone participating in the voting on the P2P (Peer to Peer) distributed network. So voting can be done transparently. Also, in the blockchain-based online voting system, voting values are connected together using the

keys and hash functions of other voting values. So it is impossible to arbitrarily modify or omit them. Since one-way calculation is easy, but inverse calculation is very difficult, the input value cannot be inferred or calculated no matter what method is used. So it is difficult to forge or alter them, and voting can be conducted transparently.

Chapter 3

Proposed System

3.1 Exploring the Existing Voting System

The existing voting system in our college comprises the central server called the admin which controls the entire voting process. Every student is eligible to vote using his/her enrollment number and the IP address for the system which is produced that enables the student to cast the vote. Also a list of passwords is prepared and distributed to the faculties handling the process which can easily be exploited by an adversary.

Drawbacks to the Existing System

- The administration holds the entire record ie.all the votes cast hence if the system runs out of order the entire process fails.
- The system fails to handle a large number of entries simultaneously. There have been many cases where the system could not record the votes and the votes got wasted.
- The password for casting a vote is generated using the students date of birth and along with the IP address of the system used for casting votes. However the password is not confidential. In Fact a list of students id and password is circulated among the people which are assigned duties during the voting process. So any anonymous person can cast votes using someone else's identity.
- The voting system requires the entire college to be present for casting the votes. Many day scholars do not feel the need to come to college just to cast their votes.
- The atmosphere during the voting day is quite tensed with different people trying their best to get more votes. Many students in order to avoid such extreme conditions do not prefer voting.
- Since there is no alternate copy of the data recorded if data is altered there will be no one that can be aware of the changes made.

3.2 Proposed System

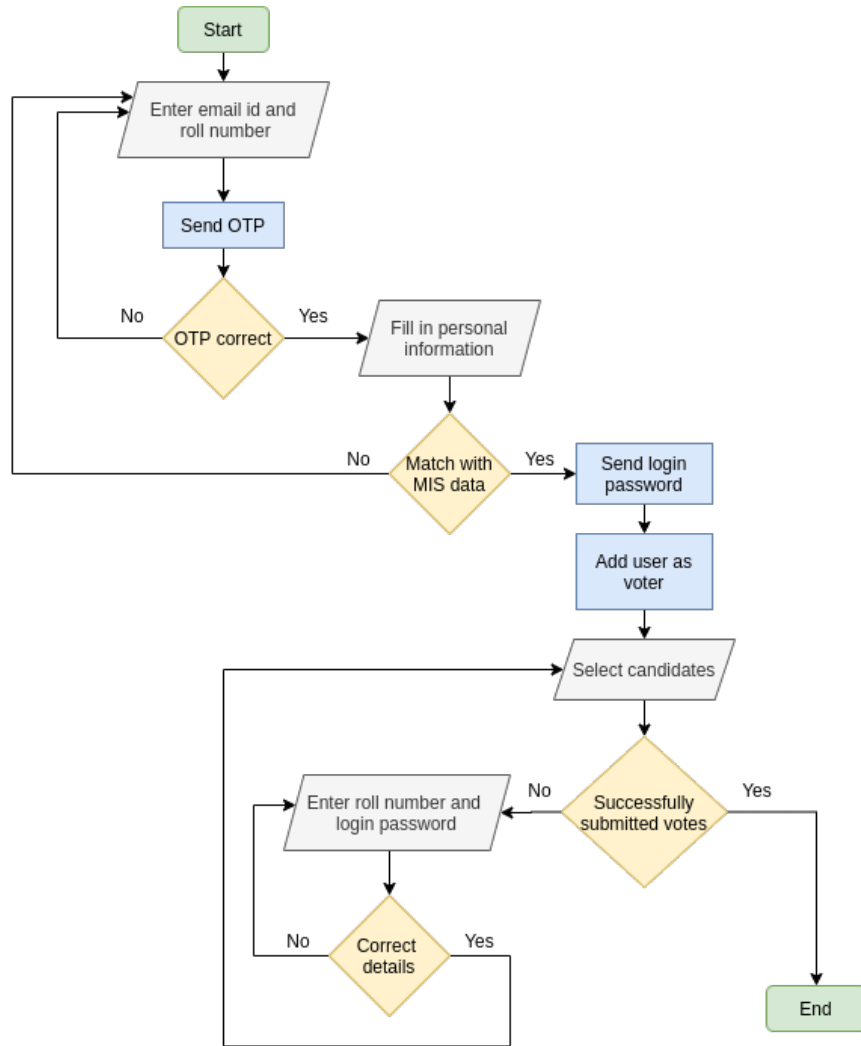


Figure 3.1: Proposed system flowchart

The flowchart in the Fig 3.1 depicts the flow in perspective of the user. A user will enter their respective email ID and roll number. An OTP will be sent to them. If the OTP is correctly imputed, they will be asked to fill in their personal information. If the OTP the user has entered is incorrect or if they have filled their personal information wrong, the system will automatically not allow them to pass to the next stage. The user will be again asked to enter their email ID and roll number. If both OTP entered is correct and the personal information entered is valid (ie. it will be verified using MIS data), the user will be sent their login in password to the email ID registered. After which, the user will be added as a voter. After that they are able to cast the vote on the blockchain network. Once they select their candidates and are able to successfully submit the votes, the system will log them out. If it was unsuccessful, the system will ask them to enter their roll number and login password. Once that is verified, they will be able to again select candidates and cast the vote.

One of the main objectives of building an online-voting system using blockchain is decentralization. In other words, being able to run the application on multiple peers gives rise to various perks. During the process, the voter is first verified before registering to vote. After the vote has been casted the data is transferred to the orderer (it can be SOLO or a cluster) which compiles the transaction and the world state variable, forming the block to be appended to the chain. The peers used in this application are the committing peers which do not endorse the transaction but commit the block to the chain. So after the orderer creates a block it is transmitted to the committing peers for adding blocks of transactions to the shared ledger and updating the state variables.

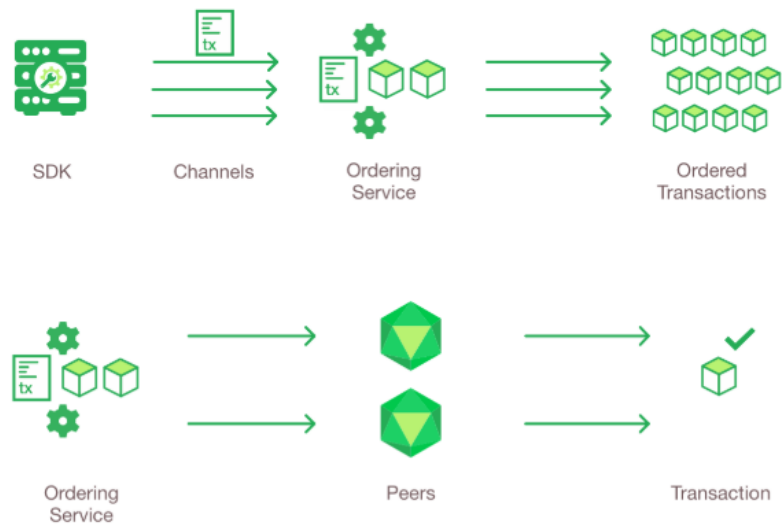


Figure 3.2: Transaction Flow [29]

3.2.1 Why are we opting Hyperledger over Ethereum?

The innovation of fabrics introduced a whole new take on the blockchain architecture. It was directed at improving its confidentiality, resilience, scalability and reliability. Designed as a modular and extensible general-purpose permissioned blockchain, Fabric is the first blockchain system to support the execution of distributed applications written in standard programming languages, in a way that allows them to be executed consistently across many nodes, giving impression of execution on a single globally-distributed blockchain computer [?].

Fabrics happened to be the first distributed operating system in the permissioned blockchain network. The architecture of the hyper fabrics follows the execute-order-validate format. This is the path traditionally followed in any distributed execution of code that users do not trust, in an unsafe environment. The transactional flow is segregated into three phases, which can run on different entities within the system:

- (a) Endorsement- endorsing a transaction, checking its accuracy and validating it (corresponding to the execution of transaction).
- (b) Ordering- ordering through a consensus protocol, regardless of any trans-action semantics
- (c) Validation - transaction validation takes place that is targeted to applications. This boasts the trust assumptions and wipes out any race condition states occurring due to concurrency.

3.3 Modular Design

The entire system is to be divided into 4 sub-modules. These modules are responsible for carrying out particular tasks of the system. The following functionalities are carried out by the sub-modules.

- (a) Verification Module: The first phase of the online voting system is the verification process. This module verifies the details filled during the process of registration with the details of the student in MIS data. The verification process verifies the OTP sent to the registered email id along with other details in the moodle account.

Authentication is when an entity proves an identity. We have implemented a multi step authentication. the roll number of the student is checked in the MIS database. OTP is sent to their email ID to confirm that the student has provided the correct details. Voting related information will be sent to this email id. Personal details are crossed checked from MIS database (dummy database in the project demonstration) If someone performs identity spoofing, then they are bound to fail at any one of the 3 steps.

- (b) On-boarding Module: The second step in the online voting process is on-boarding. It means after the students' details have been completely verified using the MIS data hence the students need to be linked to the application in order to vote(or perform the transaction). This is done by sending a secret key to the registered email-id the of the student which is used to login into the system.The secret key is an 8 digit random key created

using the Sodium Library(is a powerful cryptography library) which provides various set of functions to generate unpredictable data hence is very effective in order to create secret keys.

It is important to create a login page instead of just redirecting the voters to the page for casting vote because in case any power or system failure occurs and the student is unable to vote he/she can again use the login to vote .Also if any student tries to perform voting more than once he/she would not get the access to vote again.

During the On-boarding process the voters are created as participants in the blockchain with their id ,name and enrollment number.Hence on successful login the voters can perform the voting(transaction).

- (c) Casting the Vote s: On-boarded students can cast their votes during the election period. After the successful verification the voters can cast their votes. The administrative nodes including peers, add the votes onto the blockchain hence making their votes secure and immutable. Their votes along with their admission number and timestamp are sent as input to the hashing algorithm to the corresponding hash to be sent to the next block.
- (d) Declaration of Results: The admins will be able to receive the count of votes after the ending of voting procedure.

Chapter 4

Implementation Methodology

We have followed a similar systematic approach to building and developing the back end of the blockchain voting model. Throughout this chapter, code snippets and screenshots of the working model will be presented.

A Gantt chart pictorially describes various tasks in relation to each other with a defined timeline view [20]. The entire development of the system can be seen via the gantt chart shown in fig. It is a tremendous help in project management, thus making the implementation methodology easier.



Figure 4.1: Gantt Chart

4.1 Peer Network

Currently, we have built a network comprising 2 peers, wherein there is an admin node and a peer node. The admin node also serves as the orderer, certificate authority and a peer itself. Below we have added some screenshots of the working model. One can see code snippets depicting the above-mentioned theory.

```

6  version: '2'
7
8  networks:
9    basic:
10     services:
11       ca.example.com:
12         image: hyperledger/fabric-ca
13         environment:
14           - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
15           - FABRIC_CA_SERVER_CA_NAME=ca.example.com
16           - FABRIC_CA_SERVER_CA_CERTFILE=/etc/hyperledger/fabric-ca-server-config/ca.org1.example.com-cert.pem
17           - FABRIC_CA_SERVER_CA_KEYFILE=/etc/hyperledger/fabric-ca-server-config/8b2c293dfabf379eb91968659b248e6e3b994188ea72c6ec0
18         ports:
19           - "7054:7054"
20         command: sh -c 'fabric-ca-server start -b admin:adminpw'
21         volumes:
22           - ./crypto-config/peerOrganizations/org1.example.com/ca:/etc/hyperledger/fabric-ca-server-config
23         extra_hosts:
24           - "peer1.org1.example.com:172.21.1.211"
25         container_name: ca.example.com
26         networks:
27           - basic
28
29       orderer.example.com:
30         container_name: orderer.example.com
31         image: hyperledger/fabric-orderer
32         environment:
33           - FABRIC_LOGGING_SPEC=info
34           - ORDERER_GENERAL_LISTENADDRESS=0.0.0.0
35           - ORDERER_GENERAL_GENESISMETHOD=file
36           - ORDERER_GENERAL_GENESISFILE=/etc/hyperledger/configtx/genesis.block
37           - ORDERER_GENERAL_LOCALMSPID=ordererMSP
38           - ORDERER_GENERAL_LOCALMSPDIR=/etc/hyperledger/msp/orderer/msp
39         working_dir: /opt/gopath/src/github.com/hyperledger/fabric/orderer
40         command: orderer
41         ports:
42           - "7050:7050"
43         volumes:
44           - ./config:/etc/hyperledger/configtx
45           - ./crypto-config/ordererOrganizations/example.com/orderers/orderer.example.com:/etc/hyperledger/msp/orderer
46           - ./crypto-config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com:/etc/hyperledger/msp/peer0rg1
47         extra_hosts:
48           - "peer1.org1.example.com:172.21.1.211"
49         networks:
50           - basic
51

```

Figure 4.2: Docker-compose.yml file

YAML is a human-readable data-serialization language. It is commonly used for configuration files and in applications where data is being stored or transmitted. With docker-composer, we use YAML to configure our applications services. As can be seen in lines 12 and 29, we will create a multiple-container docker configuration wherein fabric-ca, orderer, peer and couchdb services will be isolated from each other.

Along with creating the configuration, this file also provides the location of the CA. The CA filename is updated every time the server is generated and the filename needs to be updated here in the docker-compose.yml file according to the filename in the respective directory. This can be seen in Fig 4.2. Additionally, in order for the peers to be able to connect to the network, their IP address needs to be written under the extra hosts section of the code above (172.21.1.211 is the IP of the second peer, peer1.org1.example.com).

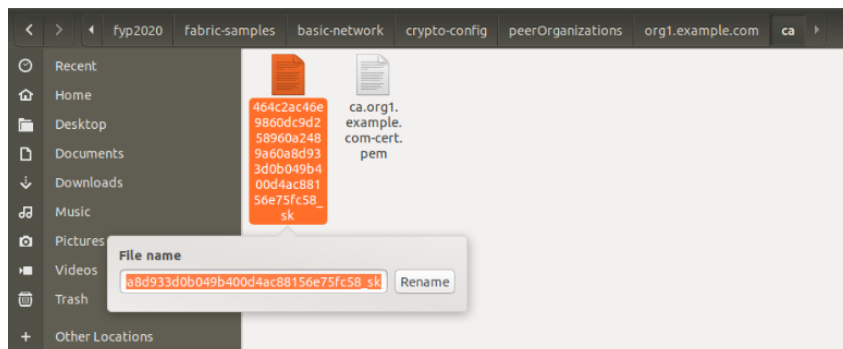


Figure 4.3: CA file for peer authentication

Start.sh script first removes the previous instances of all the peers and orderer and then creates new docker containers, one container for each image. All the services are listening at different

ports and will later help in providing authentication, database, transaction ordering facility and to establish communication with the client. The unique docker container IDs and status of the same can be seen in figure

```
docker-compose -f docker-compose.yml up -d ca.example.com orderer.example.com peer0.org1.example.com couchdb
Creating couchdb ... done
Creating peer0.org1.example.com ... done
Creating ca.example.com ...
Creating couchdb ...
Creating peer0.org1.example.com ...
docker ps -a
```

CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
3267d95cd04f2	hyperledger/fabric-peer	peer0	"peer node start"	5 seconds ago	Up	Less than a second
051/tcp, 0.0.0.0:7053->7053/tcp		peer0.org1.example.com				
140b1fa5a9d2	hyperledger/fabric-couchdb	couchdb	"tini -- /docker-ent..."	15 seconds ago	Up	5 seconds
tcp, 9100/tcp, 0.0.0.0:8050->8050/tcp		couchdb				4369/tcp, 5984/tcp
249ae20af34f	hyperledger/fabric-ca	ca	"sh -c 'fabric-ca-se..."	15 seconds ago	Up	6 seconds
054/tcp		ca.example.com				0.0.0.0:7054->7054/tcp
e3dcd017874b	hyperledger/fabric-orderer	orderer	"orderer"	15 seconds ago	Up	6 seconds
050/tcp		orderer.example.com				0.0.0.0:7050->7050/tcp

Figure 4.4: Server Started

Moving on to the configuration of the second peer, peer1.org1.example.com. This peer (peer1.org1.example.com) is using port 8051 to communicate with the server residing (peer0.org1.example.com) at port 7051. Extra hosts are the server machine which also serves as the orderer and certificate authority providing authentication details to all the peers wanting to connect to the network. The IP address 172.16.2.237 is of the server machine. Similar to the server machine the contents of docker can be seen in figure 4.5

```

39 ports:
40   - 8051:7051
41   - 8053:7053
42 volumes:
43   - /var/run:/host/var/run/
44   - ./crypto-config/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/msp:/etc/hyperledger/msp/peer
45   - ./crypto-config/peerOrganizations/org1.example.com/users:/etc/hyperledger/msp/users
46   - ./config:/etc/hyperledger/configtx
47 extra_hosts:
48   - "orderer.example.com:172.16.2.237"
49   - "peer0.org1.example.com:172.16.2.237"
50   - "ca.example.com:172.16.2.237"
51 depends_on:
52   - couchdb1
53   # - orderer.example.com
54   # - peer0.org1.example.com
55 networks:
56   - basic
57 couchdb1:
58   container_name: couchdb1

```

Figure 4.5: Dockercompose.yml file for peer1

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
91072c5fe069	hyperledger/fabric-peer	"peer node start"	6 minutes ago	Up 6 minutes	0.0.0.0:8051->7051/tcp, 0.0.
0.0:8053->7053/tcp	peer1.org1.example.com				
d4cd3c1ef342	hyperledger/fabric-couchdb	"tini -- /docker-ent..."	6 minutes ago	Up 6 minutes	4369/tcp, 9100/tcp, 0.0.0.0:
6984->5984/tcp	couchdb1				

Figure 4.6: Docker containers status in peer1

Hyperledger Fabric supports two types of peer databases: LevelDB is the default state database embedded in the peer node and stores chaincode data as simple key-value pairs; and CouchDB is an optional alternate state database that supports rich queries when chaincode data values are modeled as JSON. CouchDB is a JSON document datastore rather than a pure key-value store, therefore enabling indexing of the contents of the documents in the database. This is why couchdb is also in one of the docker containers to maintain the world state database.

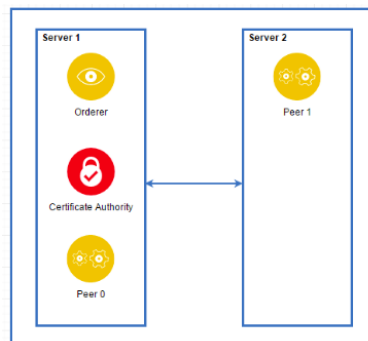


Figure 4.7: Roles of peer machines

One of the advantages of our network connection is that the peers and server stay active till there is a stable internet connection and the machines are turned on.

CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
ae07852896ff	hyperledger/fabric-peer		"peer node start"	44 hours ago	Up 44 hours	0.0.0.0:7051->7051/
tcp, 0.0.0.0:7053->7053/tcp	peer0.org1.example.com					
1b98d4099237	hyperledger/fabric-ca		"sh -c 'fabric-ca-se-"	44 hours ago	Up 44 hours	0.0.0.0:7054->7054/
tcp	ca.example.com					
bc941c2b9f6b	hyperledger/fabric-couchdb		"tini -- /docker-ent..."	44 hours ago	Up 44 hours	4369/tcp, 5984/tcp,
9100/tcp, 0.0.0.0:8050->8050/tcp	couchdb					
3c4e0d1c5a16	hyperledger/fabric-orderer		"orderer"	44 hours ago	Up 44 hours	0.0.0.0:7050->7050/
tcp	orderer.example.com					

Figure 4.8: Docker ps status

4.2 Blockchain

```
namespace org.example.empty
asset Vote identified by voteID {
  o String voteID
  --> Candidate[] candidate
}
participant Voter identified by voteID {
  o String voteID
  o String password
  o String rollNumber
  o String fullName
}
participant Candidate identified by candidateId {
  o String candidateId
  o String post
  o String firstName
  o String lastName
}
transaction VoteLog {
  o String voteID
  --> Candidate[] candidate
}
```

Figure 4.9: Model file

The file shown in fig 4.8 defines the structure of the voting business network. The model.cto file has 2 participants, one asset and one transaction definition. Voter and Candidate participants contain parameters for relevant information which will be used for verification of voters identity and for the submission of votes. Since an asset is anything of value, the Vote asset here is a registry of all the votes that have been cast by a Voter with voteID (as denoted by String voteID parameter) to a list of candidates of different posts. Lastly, VoteLog is the immutable record of vote transactions which contains the parameters as Vote asset, along with timestamp and transactionID. This can be seen later in the report.

```

/**
 * @param {org.example.empty.VoteLog} tx
 * @transaction
 */
async function logAdd(tx) {
  let assetRegistry = await getAssetRegistry('org.example.empty.Vote');
  var factory = getFactory()
  var asset = factory.newResource('org.example.empty', 'Vote', tx.voteID)
  asset.candidate = tx.candidate
  await assetRegistry.add(asset);
}

```

Figure 4.10: Logic.js file

The Login.js file, shown in fig 4.9 contains the transaction logic these functions are called transaction processor functions. It defines what actions are to be performed before the vote can be submitted in the ledger. The above code snippet implements the functioning of adding the vote as an asset before it is submitted as a transaction. Updations to vote assets can be used by voters or participants in the business network to check whom they have voted for without the need to extensively search in the whole transaction log. Transaction processor functions use async functions for promises to be resolved before committing the transaction.

```

/**
 * Access control rules for vote-network
 */
rule Default {
  description: "Allow all participants access to all resources"
  participant: "ANY"
  operation: ALL
  resource: "org.example.empty.*"
  action: ALLOW
}

rule SystemACL {
  description: "System ACL to permit all access"
  participant: "ANY"
  operation: ALL
  resource: "org.hyperledger.composer.system.*"
  action: ALLOW
}

```

Figure 4.11: Permissions.acl file

As suggest by the second line in the code snippet in fig 4.10, these are the access control rules for the vote-network. These rules defines what role is played by each identity in the network on different resources. The Default rule states that any participant is allowed to access are operations on all resources. Resources here mean participants (candidate and voter), assets (vote), and transaction (VoteLog). Hyperledger Composer system namespace, in rule SystemACL is the base definition of all business network class definitions. All asset, participant, and transaction definitions extend those defined here. SystemACL is permitting all access.

```
File Edit View Search Terminal Help
Card file: /tmp/PeerAdmin@hlfv1.card
Card name: PeerAdmin@hlfv1

Command succeeded

The following Business Network Cards are available:
Connection Profile: hlfv1
```

Card Name	UserId	Business Network
admin@vote-network	admin	vote-network
PeerAdmin@hlfv1	PeerAdmin	

Figure 4.12: The cards used in the system

After setting the foundation of our business network, the next essential task is the generation of cards. In our vote-network, there are two cards PeerAdmin@hlfv1 and admin@vote-network. The PeerAdmin is used to administer the local Hyperledger Fabric and its a special role reserved for deploying business networks as well as creating, issuing, and revoking ID cards for business networks. So, with the help of PeerAdmin card, admin@vote-network is generated. Using this card, the running business network will be updated and querying the various registries (participant, identity, and so forth).

4.3 REST Server API

Once the business network is deployed locally, there must be an interface which can help a secondary user use all the operations related to a participant, an asset or a transaction. Composer REST server api serves this purpose by bridging the gap between a client application and blockchain. The default port number used by composer rest server is 3000 and it runs on localhost. Any request made to the business is done via REST server by calling the api and different functions in a jquery script over at the client web application. The usual operations are get, post, delete, etc. These are also visible in the figure below where Candidate participant operations are listed in the REST server explorer.

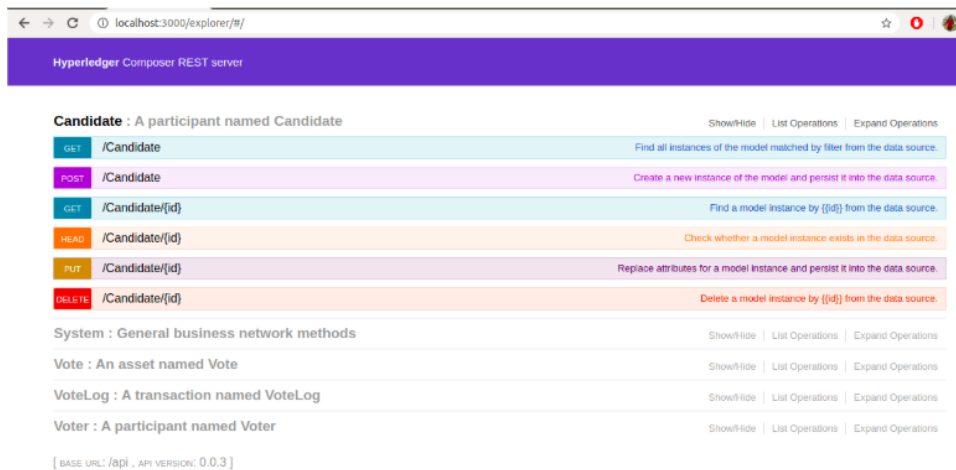


Figure 4.13: Composer REST server explorer

4.4 Client-Side Application

As depicted in the proposed flow, the first step is for the voter to enter their roll number and email id, so that an email can be sent to them with OTP. Once the OTP sent is matched with the OTP typed in the next page by a student. The student is redirected to personal details verification. As can be seen in the figure below, the students mothers name, fathers name and date of birth are asked. Details like mothers name and fathers name are not even known to the closest of friends. Ideally, these details will be checked with the actual MIS database of the college. However, we have used dummy data provided by fellow batch mates to aid the demonstration of the verification process.

NATIONAL INSTITUTE OF TECHNOLOGY, SURAT

Registration Login

Register yourself here
for the upcoming
election!

Registration

Full Name
Enter your Fullname

Mobile No.
Enter your mobile number

DOB
YYYY-MM-DD

Father's Name
Enter your father's name

Mother's Name
Enter your mother's name

Admission No.
Enter your Admission number

Figure 4.14: Details verification portal

NATIONAL INSTITUTE OF TECHNOLOGY, SURAT

Your votes Logout

Welcome U16CO116, vote here!

Student General Secretary

Pratik Kulkarni

Pratik Kulkarni

Pratik Kulkarni

Cultural Affairs Secretary

Figure 4.15: Voting Page

After the details entered by the voter are successfully verified against the MIS data, the students roll number and a few other parameters are added in the Voter participant, wherein a voteID is also generated by compiling personal details and generating a SHA-256 hash. A unique password is also generated and mailed to the student. The student is then redirected to the voting page. In the voting page, names of all the candidates will be displayed according to the posts they will be contesting for and each student will have to vote for every post and only then will the votes be added to the transaction. In the event of a bad internet connection where the student may be unable to vote, they must use the login details sent to them to login and cast their votes.

```

{
  "$class": "org.example.empty.VoteLog",
  "voteID": "0001",
  "candidate": [
    "resource:org.example.empty.Candidate#1906",
    "resource:org.example.empty.Candidate#3891",
    "resource:org.example.empty.Candidate#5686",
    "resource:org.example.empty.Candidate#7832"
  ],
  "transactionId": "e68884d8-fb04-4c31-af75-cbe367c3cd16",
  "timestamp": "2020-06-02T05:06:33.198Z"
}

```

Figure 4.16: Backend once vote is cast for a sample transaction

Once the users have submitted their votes for every post using their voteID, the candidate roll numbers will be sent in a candidate array. The sample transaction can be seen in the figure above, where the array is followed by transactionID and timestamp.

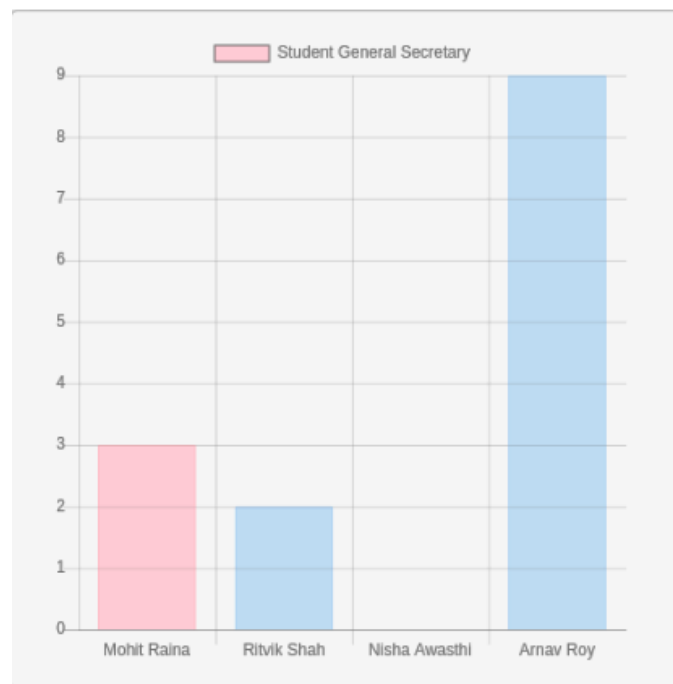


Figure 4.17: Results representation

After the voting is completed, all the admins and peers will have to bring down the composer rest server. Once this is done, after a pre-decided time interval, a button to view the results will become active and the admin will be able to view the results. The results for each post will be visible in graph format as shown in fig 4.16

Chapter 5

Results and Analysis

Upon completion of the project, an in depth analysis was done. The foremost question that had to be answered was how our project overcomes the drawbacks of the previous system and how blockchain solved these difficulties. The key areas where the analysis focused on were as follows:-

DECENTRALIZATION The proposed system provides a decentralized online voting application which is not monitored by a singular authority. All the votes will be submitted to different peers, all of which will be connected to the rest server. If anyone with malicious intent will try to alter the votes, they will have to do the same for all the peers in the network as each peer will contain a local copy of the ledger. Therefore, the server to which all requests are made is reliable and trustworthy. The more the number of peers, the better the decentralization. However, this would also require more storage in each peer machine.

IMMUTABILITY Immutability is defined as the ability of distributed ledger to remain unchanged. Firstly, this voting network allows votes to be cast only once per voteID. If a transaction is submitted with a previously used voteID through hacking the client application, then it is denied by the system. Hence, once submitted, nobodys votes can be changed. In order for an outsider to orchestrate a false vote, the attacker first needs to have the hash of the previous block, only then can he/she attempt to submit a vote. Moreover, since the network is distributed to multiple peers, if a change is successful given the condition above is satisfied, then the change must be transmitted to all the peers and the blocks following it.

PRIVACY (OF VOTES) Votes are submitted using unique voteIDs which are generated using a cryptographic hashing algorithm. Hence, when the votes are added to vote asset and transaction log, even the peers will not be able to understand as to which students has cast a particular vote.

AUTHENTICATION This online voting application has the primary goal of allowing students to vote regardless of their location. Therefore, it is increasingly important that the correct authentication schemes are applied. In this 2 step verification, email ids of students are verified through otp verification and then their details are checked against MIS data. In the current system, all authentication related information is stored in a database with the admin, but in this case, only the login password is stored in the blockchain which is also encrypted. So unless someone knows the components of their password, they cannot login into any student's account to vote.

ILLEGAL CHECKING OF VOTES Another problem significant to the current voting system is the admins ability to check vote count amidst the voting process. This leads to practices where students are told by other students to vote for person A who is x votes behind person B. To overcome this problem, the button used to count the votes is only active after the composer rest server is brought down for a certain amount of time. This preserves the fundamental attribute of any voting process.

Apart from the analysis on how well it overcomes some of the existing problems faced by the college voting system, the implementation of blockchain technology brings along with it some new challenges and drawbacks.

Drawbacks

- **Server crash**

In the event of the administration system crashing i.e. shutting down, the whole peer network will come down and will not function as admin is the also the orderer and ca server. However, if any of the peers stop working it will not affect the functioning. In any case, world state database will still have the most recent transactions stored.

- **Fixing faults**

In the event of any faults occurring, it will be difficult for anyone other than a person having blockchain knowledge to fix a problem

Chapter 6

Conclusion and Future Work

The idea of adapting and reshaping our college voting system to make the public election process cheaper, easier and quicker is quite a gripping one. The addition and implementation of blockchain technology opens the door for a more direct style of Democratic system. It allows voters to know the value of their vote, make sure their voice is heard and that their vote is not going to be tampered with. In this paper, we have implemented a unique, hyperledger blockchain system that utilizes chaincode to enable efficient and secure mechanisms to cast a vote. We have outlined the proposed flow of the system, the systems architecture, the design, and working implementation of said system. By comparison to the previous system, this new voting method overcomes a lot of the disadvantages faced by college students during the election period. On a demo run election, the results, the working and flow of the entire system depicted a good output. We were unable to use it in the elections that were supposed to take place in April due to the recent pandemic, but if given the chance would appreciate seeing how well our system does on such a large scale. According to our plan, the college elections would have been done using our blockchain based voting system for the upcoming 2020-2021 batch. Nonetheless, our primary aim of bringing more transparency to the system and expanding voter participation is achieved via our application.

In the future, as stated in the prior paragraph, it would be extremely helpful for us to see how our application stands once students of our college use it to cast the votes. Whether or not any bugs, flaws, or problems arise once 3000+ users enter the system and cast the vote. If given more time and resources, our group would want to use an authentication platform instead of OTP/MIS details for verifying a students identity. Platforms such as Google OAuth should integrate well with our system.

The present application does not use any third party for authorization instead creates passwords for the authenticated voters. However this process of giving access to the students can be improvised using the OAuth framework. It enables a third party application (client) to obtain access with specific permissions to a protected resource, with the consent of the resource owner [27]. Access to the resource is achieved through access tokens. JWT is an open standard that defines a compact format to transmit claims between parties as a JSONObject.

An extension to the current implementation can be binding identities with participants. Identities are provided in the form of cards which are generated and issued by the network admin when a participant is added to the network. These cards are stored in the local wallets of network admin and composer performs participant to identity mapping. So, whenever a participant wants to access resources, their identity card is used.

Usually, participants with identities are implemented when participants are recurring users of the business and want to exchange or own certain assets. That being said, they do provide a more controlled access to operations as permissions defining access can be altered for each type of participant. In the proposed voting application, voter participant is used to merely store student information for use during login activity and providing voteID. However, identities can be used for a more defined access control.

Bibliography

- [1] J. Göbel, H. Keeler, A. Krzesinski and P. Taylor, "Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay", *Performance Evaluation*, vol. 104, pp. 23-41, 2016. Available: 10.1016/j.peva.2016.07.001.
- [2] "e-voting system using blockchain", *International Research Journal of Engineering and Technology*, vol. 6, no. 1, 2020. Available: <https://www.irjet.net/archives/V6/i1/IRJET-V6I1322.pdf>. [Accessed 15 May 2020].
- [3] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, ... S. Muralidharan, (2018, April). Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference* (pp. 1-15).
- [4] Architecture Origins, hyperledger. [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-1.4/arch-deep-dive.html>. [Accessed: 15-Dec-2019].
- [5] A. Ahmed Ben. "A conceptual secure blockchain-based electronic voting system." *International Journal of Network Security Its Applications* 9, no. 3 (2017): 01-09.
- [6] Y. Emre, A. Kaan Ko, U. Can abuk, and G. Dalkl. "Towards secure e-voting using ethereum blockchain." In *2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, pp. 1-7. IEEE, 2018.
- [7] Hjálmarsson, Fririk and Hreiðarsson, Gunnlaugur K and Hamdaqa, Mohammad and Hjálmtýsson, Gísli. "Blockchain-based e-voting system." In *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pp. 983-986. IEEE, 2018.
- [8] K. Kashif Mehboob, J. Arshad, and M. Mubashir Khan. "Secure digital voting system based on blockchain technology." *International Journal of Electronic Government Research (IJEGR)* 14, no. 1 (2018): 53-62.
- [9] W. Jan Hendrik. "The Blockchain: a gentle four page introduction." *arXiv preprint arXiv:1612.06244* (2016).
- [10] H. Freya Sheer, A. Gioulis, R. Naeem Akram, and K. Markantonakis. "E-voting with blockchain: An e-voting protocol with decentralisation and voter privacy." In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 1561-1567. IEEE, 2018.

- [11] Barnes, Andrew, Christopher Brake, and Thomas Perry. "Digital Voting with the use of Blockchain Technology." Team Plymouth Pioneers-Plymouth University (2016).
- [12] K. Fung, Network security technologies. Boca Raton, FL: Auerbach Publications, 2005, p. 14.
- [13] A. Jadhav, "What is Ethereum Blockchain and how it works and what it can be used for?", EDIT., 2019. [Online].[Accessed: 03- May- 2020].
- [14] S. Kumar, "The Ultimate Guide to Consensus in Hyperledger Fabric — Skcript", Skcript, 2018. [Online].[Accessed: 22- May- 2020].
- [15] T. Kuhrt, "Hyperledger Fabric - Hyperledger Fabric - Hyperledger Confluence", Wiki.hyperledger.org, 2019. [Online]. [Accessed: 17- May- 2020].
- [16] S. Karim, U. Ruhi, and R. Lakhani. "Conceptualizing blockchains: characteristics applications." arXiv preprint arXiv:1806.03693 (2018).
- [17] W. KENTON, "Hyperledger Fabric", Investopedia, 2020. [Online]. [Accessed: 03- Jun- 2020].
- [18] S. Kumar, "The Ultimate Guide to Consensus in Hyperledger Fabric — Skcript", Skcript, 2018. [Online]. [Accessed: 22- May- 2020].
- [19] "Deploying Business Networks — Hyperledger Composer", Hyperledger.github.io, 2017. [Online].[Accessed: 28- Apr- 2020].
- [20] D. Rastogi, Mind Mapping to Gantt Charts, International Journal of Scientific and Research Publications, vol. 5, no. 8, Aug. 2015.
- [21] H. Rifa, and B. Rahardjo. "Blockchain based e-voting recording system design." In 2017 11th International Conference on Telecommunication Systems Services and Applications (TSSA), pp. 1-6. IEEE, 2017.
- [22] F. Francesco, M. ILARIA Lunesu, F. EROS Pani, and A. Pinna. "Crypto-voting, a Blockchain based e-Voting System." In KMIS, pp. 221-225. 2018.
- [23] Ibm, Getting started with Blockchain, what are the steps and the pitfalls?, Medium, 11-Apr-2018. [Online]. Available: <https://medium.com/@lennartfr/getting-started-with-blockchain-what-are-the-steps-and-the-pitfalls-db07d69bad6d>. [Accessed: 26-Apr-2020].
- [24] S. Panja and B. Kumar Roy, "A secure end-to-end verifiable e-voting system using zero knowledge based blockchain", IACR Cryptol. ePrint Arch., 2018. [Accessed 21 April 2020].
- [25] Fabric CA User's Guide, hyperledger, 2017. [Online]. Available: <https://hyperledger-fabric-ca.readthedocs.io/en/release-1.4/users-guide.html>. [Accessed: 05-Oct-2019].
- [26] J. Lopes, J. Pereira and J. Varajo, "Blockchain Based E-voting System: A Proposal", Twenty-fifth Americas Conference on Information Systems, Cancun, 2019, 2019. Available: <https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1296context=amcis2019>. [Accessed 4 April 2020].

- [27] V. Siris, D. Dimopoulos, N. Fotiou, S. Voulgaris and G. Polyzos, "OAuth 2.0 meets Blockchain for Authorization in Constrained IoT Environments", IEEE 5th World Forum on Internet of Things (WF-IoT), vol. 152, pp. 364-367, 2019. Available: <https://www.semanticscholar.org/paper/OAuth-2.0-meets-Blockchain-for-Authorization-in-IoT-Siris-Dimopoulos/08bd59d5c3d778548b886f919a7782fb1e2c9ebd>.
- [28] R. Zhang, R. Xue and L. Liu, "Security and Privacy on Blockchain", ACM Computing Surveys, vol. 52, no. 3, pp. 1-34, 2019. Available: <https://arxiv.org/pdf/1903.07602.pdf>.
- [29] S. B. Mills, Blockchain Design Principles, Medium, 22-Mar-2017. [Online]. Available: <https://medium.com/design-ibm/blockchain-design-principles-599c5c067b6e>. [Accessed: 16-May-2020].
- [30] S. S. Mukhekar, Deploy Business Network Archive (.bna) files to your IBM Blockchain, BlogSaays, 13-Sep-2017. [Online]. Available: <https://www.blogsaays.com/deploy-business-network-archive-to-ibm-blockchain/>. [Accessed: 05-Feb-2020].
- [31] B. Ekici, Blockchain-Based Vote Application On Hyperledger Composer, Medium, 19-Mar-2020. [Online]. Available: <https://medium.com/coinmonks/blockchain-based-vote-application-on-hyperledger-composer-e08b1527031e>. [Accessed: 26-May-2020].

Acknowledgment

We take this opportunity to thank all the individuals who have helped us in completing our final year engineering project successfully. We would like to express our heart felt gratitude to Mr. Bhavesh N. Gohil, who as our guide/mentor, has provided us with a tremendous amount of support and guidance at every step. The project would never have been completed without his encouragement and support. We sincerely thank Dr. Mukesh A. Zaveri, Head of Department for providing us with the resources and giving us a healthy environment for carrying out our project work.

Appendix

- **Software Requirements**

- Operating System : Linux 18.04 LTS and above, mac OS X(min ver. 10.12.6)
- Web Portal : HTML, CSS, Bootstrap
- Languages : PHP, JQuery, Javascript
- Technology : Blockchain
- Packages and Frameworks : Hyperledger Fabric,Hyperledger Composer,REST API
- Extra : Docker, node(8 and above but not 9), npm 5x, Xcode(mac OSX), composer-cli@0.19

- **Hardware Requirements**

- System : Intel i5 and above, mac OS X(min ver. 10.12.6)
- Hard Disk : 1TB
- Monitor : Any
- Input Devices : Keyboard, Mouse
- RAM : 8GB

- **Execution Steps**

After installing all the prerequisites mentioned in software requirements, execute the following steps on the terminal of your linux machine [31]:

1. Go to the directory [`cd /fabric-dev-servers`]
2. Export hyperledger fabric version 1.1. [`export FABRIC_VERSION=hlfv11`]
3. `./startFabric.sh` (This instruction should return a result like in Fig 4.4)
4. `./createPeerAdminCard.sh` (This should return command succeeded)

Now, to create the business network, execute the following commands in the terminal:

1. `yo hyperledger-composer:businessnetwork`
2. You will be asked for the network name, type in `vote-network` as the network name.
Enter description author name and email on prompt.
3. Select Apache-2.0 as the license
4. Enter `org.example.empty` for the namespace

5. Select No when asked whether to generate an empty network

Following the above steps will create a directory named vote-network (business network). Go to vote-network directory and execute the next steps to define the network.

1. Go to the models directory, open models.cto and copy the code present in Fig 4.9.
2. Go to the lib directory, open logic.js and copy the code present in Fig 4.10.
3. Open permissions.acl file and copy code present in Fig 4.11.

The following commands are to be run on the terminal in the vote-network directory.

1. `composer archive create -t dir -n .`
2. `composer network install --card PeerAdmin@hlfv1 --archiveFile vote-network@0.0.1.bna`
3. `composer network start --networkName vote-network --networkVersion 0.0.1 --networkAdmin admin --networkAdminEnrollSecret adminpw --card PeerAdmin@hlfv1 --file networkadmin.card`
4. `composer card import --file networkadmin.card`

These steps will result in an output like Fig 4.12 and the next steps will generate the composer rest server.

1. `composer-rest-server`
2. Enter card name example: `admin@vote-network`.
3. Select never use namespaces when asked whether to use namespaces in the generated API.
4. Select No when asked whether to secure the generated API.
5. Select No when asked whether to enable authentication for the REST API using Passport.
6. Select No when asked whether to enable multiple user and identity management using wallets.
7. Select Yes when asked whether to enable the explorer test interface.
8. Dont enter any key when asked to enter a key for enabling dynamic logging.
9. Select Yes when asked whether to enable event publication.
10. Select No when asked whether to enable TLS security.

After these steps a explorer window will open stating Hyperledger-REST-Server

For Student verification MIS data is to be stored in database. To store this data, following things are done by admin:

1. Data is stored i

To add MIS data and Candidate data in database, following steps to be done:

- (a) Open the webpage admin.php(for adding MIS data) or adminc.php(for adding candidates)
- (b) Click on choose file and choose the csv file from PC
- (c) Once Import button is clicked the file data is imported to database

Hence, the data is added to database.

Following steps to be done to add the candidate as participant on Blockchain-REST-Server:

- (a) Open the webpage candidates.php
- (b) Once you click on the submit button all the data would be posted to the REST-Server