1. INTRODUCTION

Blockchain is the backbone Technology of Digital CryptoCurrency BitCoin. The blockchain is a distributed database of records of all transactions or digital event that have been executed and shared among participating parties. Each transaction verified by the majority of participants of the system. It contains every single record of each transaction. BitCoin is the most popular cryptocurrency an example of the blockchain. Blockchain Technology first came to light when a person or Group of individuals name ‘Satoshi Nakamoto’ published a white paper on “BitCoin: A peer to peer electronic cash system” in 2008. Blockchain Technology Records Transaction in Digital Ledger which is distributed over the Network thus making it incorruptible. Anything of value like Land Assets, Cars, etc. can be recorded on Blockchain as a transaction.

Democratic voting is a crucial and serious event in any country. The most common way in which a country votes is through a paper based system, but is it not time to bring voting into the 21st century of modern technology? Digital voting is the use of electronic devices, such as voting machines or an internet browser, to cast votes. These are sometimes referred to as e-voting when voting using a machine in a polling station, and i-voting when using a web browser. Security of digital voting is always the biggest concern when considering to implement a digital voting system. With such monumental decisions at stake, there can be no doubt about the system’s ability to secure data and defend against potential attacks. One way the security issues can be potentially solved is through the technology of blockchains.

Blockchain technology originates from the underlying architectural design of the cryptocurrency bitcoin. It is a form of distributed database where records take the form of transactions, a block is a collection of these transactions. With the use of blockchains a secure and robust system for digital voting can be devised. This report outlines our idea of how blockchain technology could be used to implement a secure digital voting system.

As it has been observed that in some cases booth capturing has been reported, the statistics have shown that the percentage of polling on the day of elections is not satisfactory as majority of population are not coming to vote and thinks it's a wastage of time. Traditionally elections were conducted using a ballot paper where voters marked the candidates of their choice and submitted the ballot papers at the voting stations. These ballot papers were then manually counted which resulted in a huge delay in the election process. Although this method of conducting elections was fairly simple, it did not scale well. Manual counting of votes wasn't practical for large scale elections and was very time consuming. Replacing this pen and paper with a new electronic system had the potential to limit fraud and make the voting process traceable and verifiable [1].

Blockchain technology offers new tools for authentication and authorization in the digital world that preclude the need for many centralized administrators. As a result, it enables the creation of new digital relationships. By formalizing and securing new digital relationships, the blockchain revolution is posed to create the backbone of a layer of the internet

1.1 MOTIVATION

I chose this topic because of the voting problem in India, In India only 66.36% of the eligible population voted in the last elections. This shows that still there are so many people who face some kind of difficulty in voting scheme which is currently in use. Blockchain technology offers new tools for authentication and authorization in the digital world that preclude the need for many centralized administrators. As a result, it enables the creation of new digital relationships. By formalizing and securing new digital relationships, the blockchain revolution is posed to create the backbone of a layer of the internet.

The Election Commission of India introduced the Electronic Voting Machines (EVMs) in the late 1990s. These machines were exclusively produced by a group of government agencies namely Bharat Electronics Limited (BEL) and Electronics Corporation of India Limited (ECIL). These devices were cheap, easy to use and had a simple design. As a result, EVMs were widely adopted throughout the country.

Even though the machines were believed to have a simple design, the actual working mechanism of the EVMs has not been made completely public. Due to this, the EVMs have been subjected to widespread criticism by various political parties who questioned their integrity and reliability. Recently there has also been increase in the allegation of electoral frauds regarding the EVMs such as the 2009 parliamentary elections. This has raised various concerns in the minds of the citizens regarding the electoral procedure in the country. As a result of this, the authorities have been forced to look for other reliable alternatives to the standard electronic voting systems.

Due to the inherent simplicity of the EVMs it is very easy to tamper the machines. Most of the attacks on EVMs are physical in nature [2]. Anyone with the basic knowledge of electronics can disturb the mechanism of the EVM which in turn could disrupt the entire election process. Recent studies have also shown that access to the EVMs for a few minutes is enough to tamper the mechanism of the machine [5]. Current day voting system faces challenges related to security of votes and usage of EVMs. The detailed security analysis of the EVMs was done was Scott Wolchok [2]. Some of the key findings from his study are discussed below:

- Corrupt hardware can be attached to the voting machine and it can be manipulated easily. This results in favouring a particular candidate at the time of voting.
- Replacing the original device with a malfunctioned one so that every time a vote is given to a candidate it goes to a particular candidate instead of the intended one.
- The EEPROM chip of the EVM can be replaced with another chip with corrupted count of votes in favor of each candidate which can be a problem to detect.

1.2 AIM AND OBJECTIVES

Aim of this project is to create an application that provides improved voting services to the voters through fast and timely and convenient voting. Some of the key objectives needed to be considered in order to develop a voting system are listed below :

- To keep an easy track of voters.
- To restrict voters from casting votes multiple times.
- To provide less capital, less effort and less labour-intensive platform.
- To provide instant poll result.
- To develop a system that is highly scalable and overall provides a structure that is more secure than any of the existing frameworks.

2. LITERATURE SURVEY

2.1 RELATED WORK

Paper Title	Paper Type	Authors	Description
Security analysis of India's electronic voting machines	17th ACM conference on Computer and Communications Security, CCS 2010	Scott Wolchok, Eric Wustrow, J. Alex Halderman, Hari K. Prasad, Arun Kankipati, Sai Krishna Sakhamuri, Vasavya Yagati	. Demonstrated two attacks, implemented using custom hardware, which could be carried out by dishonest election insiders or other criminals with only brief physical access to the machines. This case study carries important lessons for Indian elections and for electronic voting security more generally.
Secret suffrage in remote electronic voting systems	Fourth International Conference in eDemocracy & eGovernment (ICEGEG)	Adrià Rodríguez-Pérez	Proposed to approach the principle of secret suffrage against the specificities of internet voting and, instead of evaluating using traditional standards for privacy and anonymity.

Blockchain-based e-voting system	IEEE 11th International Conference on Cloud Computing (CLOUD)	Friðrik Þ. Hjálmarsson, Gunnlaugur K. Hreiðarsson, Mohammad Hamdaqa, Gísli Hjálmtýsson	This paper evaluates the potential of distributed ledger technologies through the description of a case study, namely the process of an election and implementing blockchain based application which improves the security and decreases the cost of hosting a nationwide election
A privacy-preserving voting protocol on blockchain	11th International Conference on Cloud Computing, 2018	Yuan Yuan, Yanyan Hu, Shaohua Huang, Shengjiao Cao, Anuj Chopra	Proposed a native blockchain voting protocol for peers to vote over their existing blockchain network without the need of any trusted or third party.

Table No 2.1.1 - Literature Survey

Literature review is a text of a scholarly paper, which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic. Literature reviews are secondary sources, and do not report new or original experimental work

3. PROJECT STATEMENT

Statement of Work (or SOW) is a formal document that defines the entire scope of the work involved for a vendor and clarifies deliverables, costs, and timeline. It is needed in situations where a project involves vendors and external contributors in addition to the internal project team. You usually create a SOW as part of a bid document or part of a contract.

3.1 PURPOSE

The aim of this project is “to create a secure Digital Voting System”. The main objective of our project is to provide instant poll result and also keep a track of voters. It also includes several other economic aspects like less capital, less use of man power or labor. This system also allows only registered and valid voters to vote. The software integrates various functions like registration of voter, creating the ballot, loading the ballot and casting the vote. The central feature of the system is that there will be no involvement of a third party when the voter castes the vote and the candidates receive their votes directly without any other person knowing about it which gives us the idea of decentralized apps which in this case is decentralized voting system. This system will mainly reduce the man-power and time as all the system would be automated.

For our design we tried to create a system that doesn’t entirely replace the current voting but rather integrates within a current system. We decided to do this to allow for as many different ways to vote as possible, this is so voting can be accessed by the majority of the population.

3.2 PRODUCT PERSPECTIVE

Electronic voting systems have been a topic of active research for the last few years. Researchers and authorities have been looking for robust and highly secure methods to conduct elections. The rise in electoral frauds in the last few years have raised concerns regarding the nation’s election process. The purpose of the product is to provide more secure mechanism for storing the voting transactions in a digital voting system. We have used a permissioned blockchain in order to achieve high security and reliability

. 3.3 PRODUCT FEATURES

- Provides instant poll result.
- Keeps an easy track of voters.
- Provide higher security than traditional EVMs.
- Ensures protection against double voting
- Is less capital, less effort and less labor intensive than EVM systems.
- Satisfies all the requirements to ensure fair election process.

3.4 USER CLASSES AND CHARACTERISTICS

The users of the proposed model will primarily be voters who will be casting their votes in the election. However, there are other entities as well who will be interacting with the system. They are:

- Voters: People who are going to be interacting with the system in order to cast their vote.

- Controller: Controller will be the person who is present at the controller station interfaced with the EVM to verify the credentials and ID of the Voters.
- Testers & Developers: Testers and Developers of the system will only be involved in the development stage of the product in order to identify and resolve issues in the system.
- Supervisor: Supervisor will be the onsite engineer who will be present at each voting station in order to initiate the system and resolve any real-time issues.

3.4.1 Operating Environment

The system will operate using NodeJS based application developed on ExpressJS platform and JavaScript as a scripting language, bootstrap as framework and Socket.io for real-time communication between nodes. This system can be integrated with the VVPAT in order to run in coordination with the controller system to conduct elections at large scale.

4. SOFTWARE AND HARDWARE REQUIREMENTS

Software requirements deal with defining software resource requirements and prerequisites that need to be installed, on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed. The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application.

4.1 EXTERNAL INTERFACE REQUIREMENT

External interface requirements specify hardware, software, or database elements with which a system or component must interface.

4.1.1 User Interface

The user will interact with the functional system to a minimum. Once the controller has validated the voter's credentials, he then proceeds to the VVPAT which will be running the Node-subsystem and electrically interfaced with the controller device. The voter then casts vote to the candidate of their choice.

4.1.2 Hardware Interface

The hardware requirement for us is very less. A computer system with minimum 50GB hard disk 512MB RAM. The instruction received from frontend will be converted into machine understandable by the computer and corresponding action will be performed.

4.1.3 Software Interface

The project "Digital Voting System Using Blockchain" as the name suggests makes use of an instance of blockchain that is shared amongst all the nodes in the network. The project makes use of Javascript language and ExpressJS framework for it functioning. It also uses the Websockets library for real-time communication and data sharing between the nodes.

4.2 OTHER NON-FUNCTIONAL REQUIREMENTS

A non-functional requirement (NFR) is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors.

4.2.1 Performance Requirements

In order to perform efficiently the connections needs to be proper and intact. They should be able to resist small jerks and obstacles. The network connectivity is required for the product to work, the product won’t be able to perform actively without network availability.

4.3 SOFTWARE QUALITY ATTRIBUTES

Since the project is based on the concepts of Blockchain, the system is highly secure. All the votes registered in the system are reflected at all the nodes in the network, hence in order to tamper with the system or add fraudulent transactions or change the contents of the block, all these changes will have to be reflected at all the nodes in the network. Since it is almost impossible to make changes at such a large scale simultaneously, it is safe to believe that the system is robust as far as security is concerned.

5. DESIGINING

Software design is an interactive process through which requirements are translated into a blue print for constructing the software. The design is represented at high level of abstraction, a level that can be directly translated to specific data, functional and behavioral requirements.

5.1 SYSTEM ARCHITECTURE

Fig 5.1.1 represents the architecture and configurations of the implemented system containing one single EVM. It also represents the connections between various components of the system and how they interact with each other. Functioning of each of these components is discussed in Chapter 8.

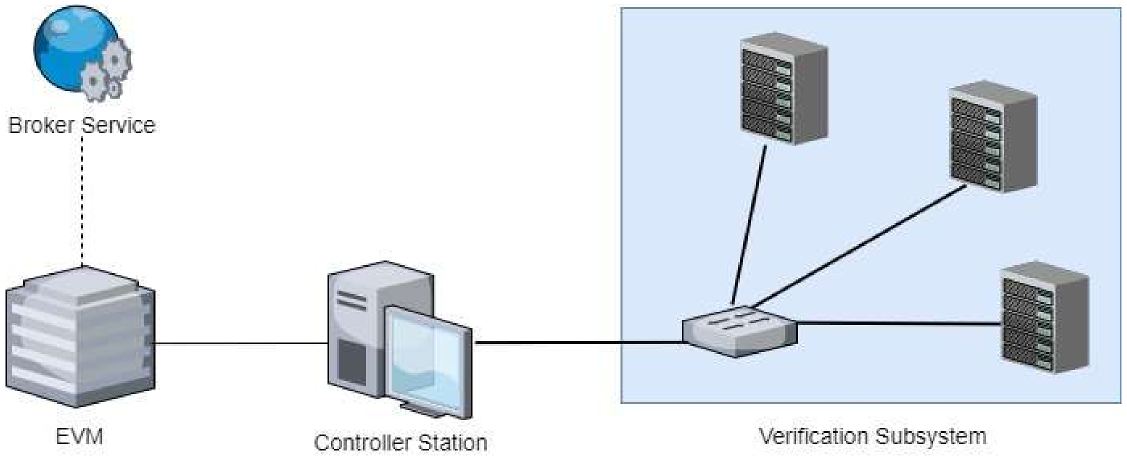


Fig 5.1.1- Architecture Of The Proposed Model

5.2 STATE DIAGRAM

A state diagram is used to represent the condition of the system or part of the system at finite instances of time. It's a behavioural diagram and it represents the behaviour using finite state transitions. State diagrams are also referred to as State machines and State-chart Diagrams. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. A state diagram is used to model the dynamic behaviour of a class in response to time and changing external stimuli. a state diagram is used to model the dynamic behaviour of a class in response to time and changing external stimuli. Fig 5.2.1 represents the state diagram showing the different states that the controller station and the EVMs exhibit during the lifetime of the election process.

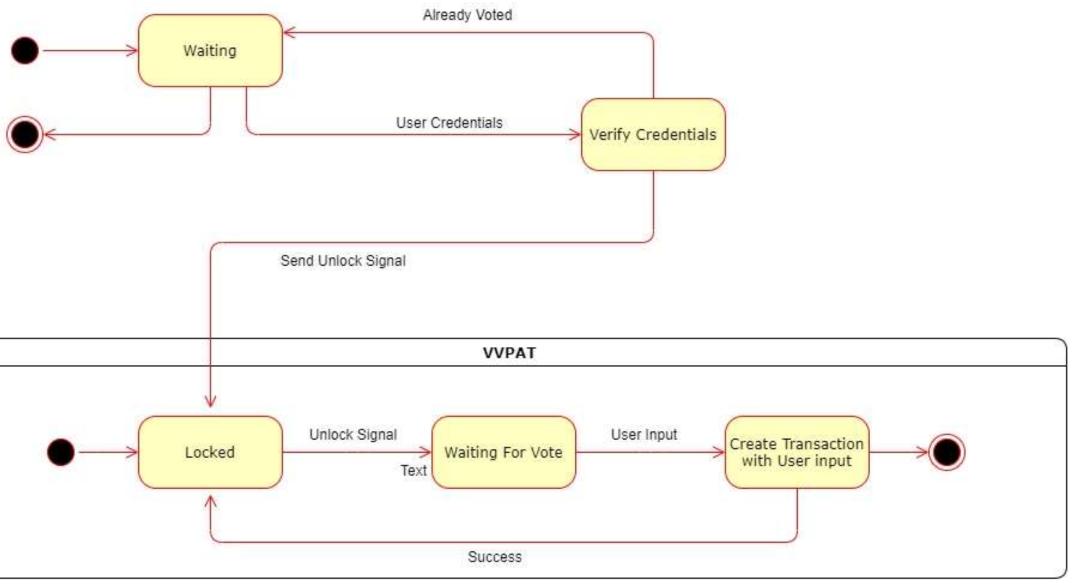


Fig 5.2.1- State Diagram for Controller And VVPAT

5.3 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities. Fig 5.2.1 represents the activity diagram. It shows start and end of all the activities taking place in the system.

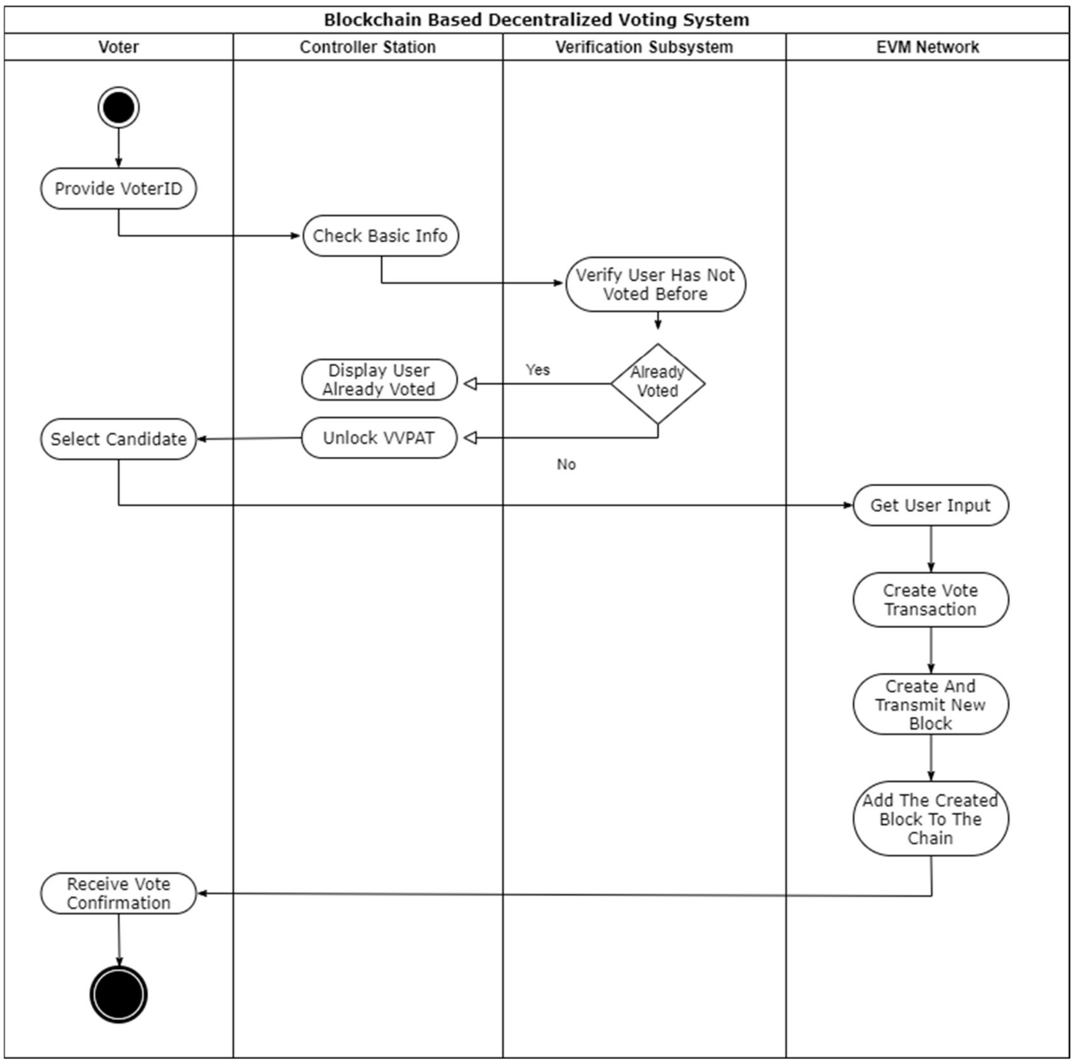

Blockchain Based Decentralized Voting System
Voter
Controller Station
Verification Subsystem
EVM Network
Provide VoterID
Check Basic Info
Verify User Has Not Voted Before
Already Voted
Yes
Display User Already Voted
No
Unlock VVPAT
Select Candidate
Get User Input
Create Vote Transaction
Create And Transmit New Block
Add The Created Block To The Chain
Receive Vote Confirmation

Fig 5.3.1- Activity Diagram

5.4 USE CASE DIAGRAM

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

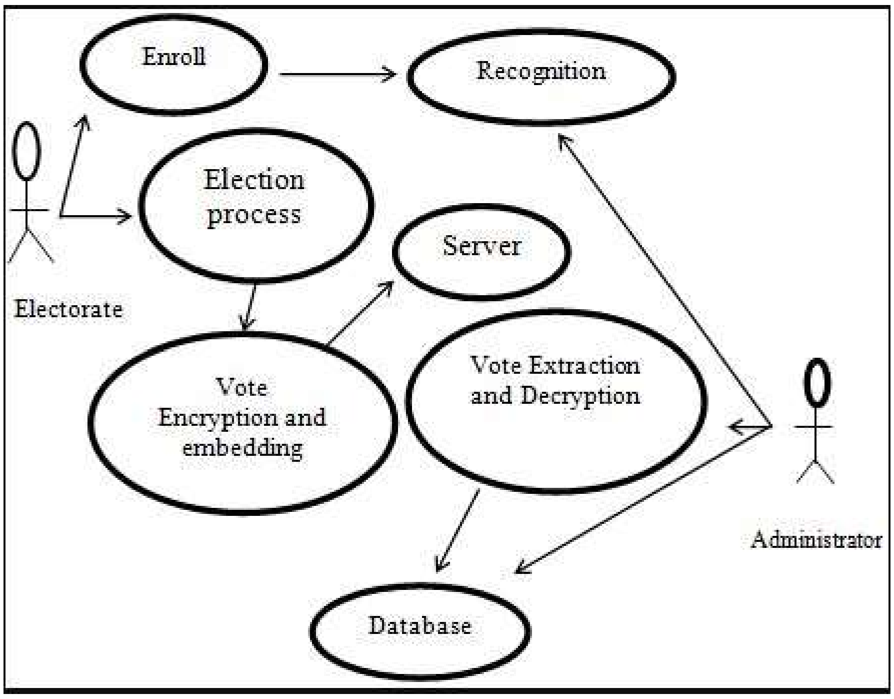


Fig 5.4.1- Use Case Diagram

6. PLANNING AND SCHEDULING

Planning refers to the action of establishing a plan, meaning a set of tasks that needs to be completed. The schedule is typically represented in a calendar or as a timeline, in chronological order. Project planning is part of project management, which relates to the use of schedules such a Gantt charts to plan and subsequently report progress within the project environment.

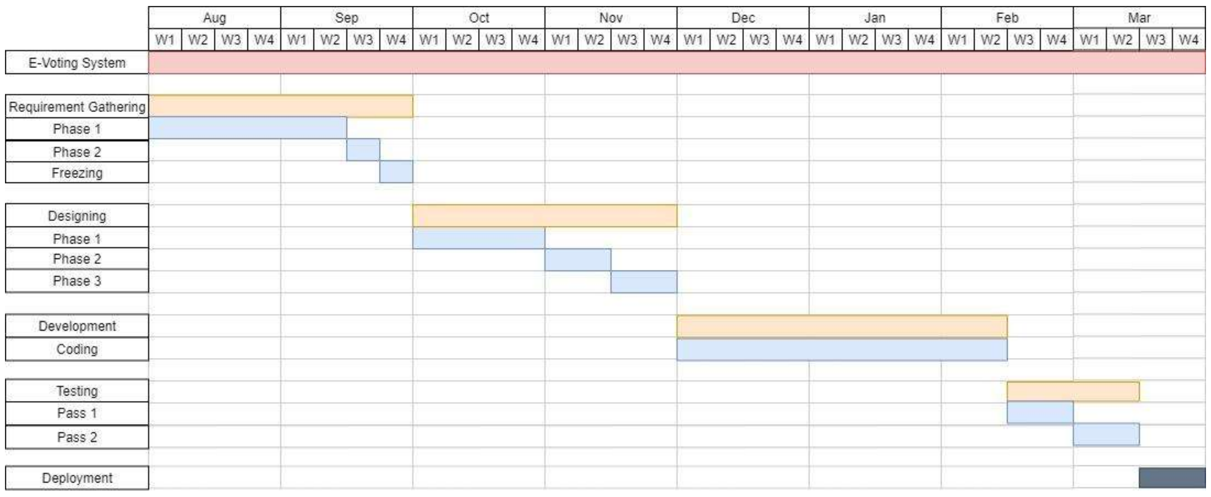


Fig 6.1- Gantt Chart for Project Scheduling

7. SOFTWARE TESTING

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing is the process of evaluation a software item to detect differences between given input and expected output. Software testing is an activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. It is more than just running a program with the intention of finding faults. Every project is new with different parameters. No single yardstick maybe applicable in all circumstances. Although critical to software quality and widely deployed by programs and testers. Software testing still remains an art, due to limited understanding of principles of software. The difficulty stems from complexity of software. The purpose of software testing can be quality assurance, verification and validation or reliability estimation. Testing can be used as a generic metric as well. Software testing is a trade- off between budget, time and quality.

7.1 TYPES OF TESTING

7.1.1 Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.1.2 Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.1.3 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.1.4 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

7.1.5 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

7.2 TEST CASES

A test case is a specification of the inputs, execution conditions, testing procedure, and expected results that define a single test to be executed to achieve a particular software testing objective, such as to exercise a particular program path or to verify compliance with a specific requirement.

SR NO.	TEST CASES NAME	TEST CASE DESCRIPTION	EXPECTED	ACTUAL
1	Broker Initialization	Ensure error-free Initialization of Broker service.	Successful	Same as Expected
2	Node Initialization	Initialize the instance of the Node on the system.	Successful	Same as Expected
3	P2P Network establishment.	Connects one node to another ensuring Peer to Peer Connection	Successful	Same as Expected
4	Check Genesis	Checks if all the nodes contain the same genesis block(Block 1).	Successful	Same as Expected

5	Check Voter Id	Traverse the blocks matching the current Voter Id.	Successful	Same as Expected
6	Cast Vote	Gather the Vote information from the System UI	Successful	Same as Expected
7	Encryption	Mask the candidate id(toID) before adding it to the block.	Successful	Same as Expected
8	Recasting Vote	Does not allows the voters to recast their vote.	Successful	Same as Expected
9	Block Tampering	Automatically replace the tampered block with the correct one in the next mining iteration.	Successful	Same as Expected
10	Closing Connections	Terminate connection to the disconnecting nodes in the network.	Successful	Same as Expected

Table No. 7.2 - Test Cases

8. IMPLEMENTATION

Project implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system through computer programming and deployment.

8.1 OVERVIEW OF THE PROJECT

We propose a reliable, efficient and highly secure way of storing the votes. In the current system, the EVMs rely on its hardware for the most part. The model proposes extended use of software in order to leverage the security features of e-voting and combine it with the robust architecture of the current system. This results in a hybrid system that is secure as well as highly scalable. However, in order to do so we need a way to connect the EVMs together in a network. This makes the EVMs vulnerable to a number of network-based attacks such as man-in-the-middle, DDoS, session hijacking, receipt stealing etc. The solution to these problems is to make use of decentralized architecture as opposed to the conventional centralized models thus eliminating the centralized points of failure. The model proposes the use of a private (permissioned) blockchain that allows only the identified EVMs to connect to the network and interact with it. This protects the system from malicious devices that may try to interfere with the voting process.

In this project we have created a barebone structure of a permissioned blockchain that can be used for storing EVM transactions and interact with other nodes in the network to successfully establish and maintain a decentralized network. Our implementation consists of the following components :

8.1.1 Verification Subsystem

The verification subsystem interacts with the voter database to verify voter's credentials. The database also consists of a status flag that indicates if the voter has previously voted or not. This helps to ensure that the voter is not allowed to vote multiple times. Initially, the status will be false or not voted for all the voters. Although a blockchain based database can be used to further improve security against double voting, the time required to search a blockchain based database increases linearly with the number of blocks. In most countries there already exists a database that contains data of its citizen, it is convenient to integrate these government databases with the system instead of creating a new database from scratch.

8.1.2 Controller Station

Each EVM works in coordination with a controller station. The controller station acts as the bridge between the verification subsystem and the EVMs. It is used to fetch the voter's information from the verification subsystem and feed the required input fields to the EVM. Its main task is to verify the voter's identity and prepare the EVM for the vote to be registered.

8.1.3 Broker Service

The broker acts as a discovery service by EVMs discover other nodes in the network and connect to them. The broker service acts as a connection manager that keeps real-time track of the nodes entering and leaving the network. Every time a new node is connected to the network, the broker service provides it with a list of all the active nodes in the network so that it can initiate connections to them. It then adds the metadata of the incoming node to the list of active nodes. When a node leaves the network, the broker notifies all the nodes in the network to terminate connections with the leaving node.

8.1.4 EVM Network

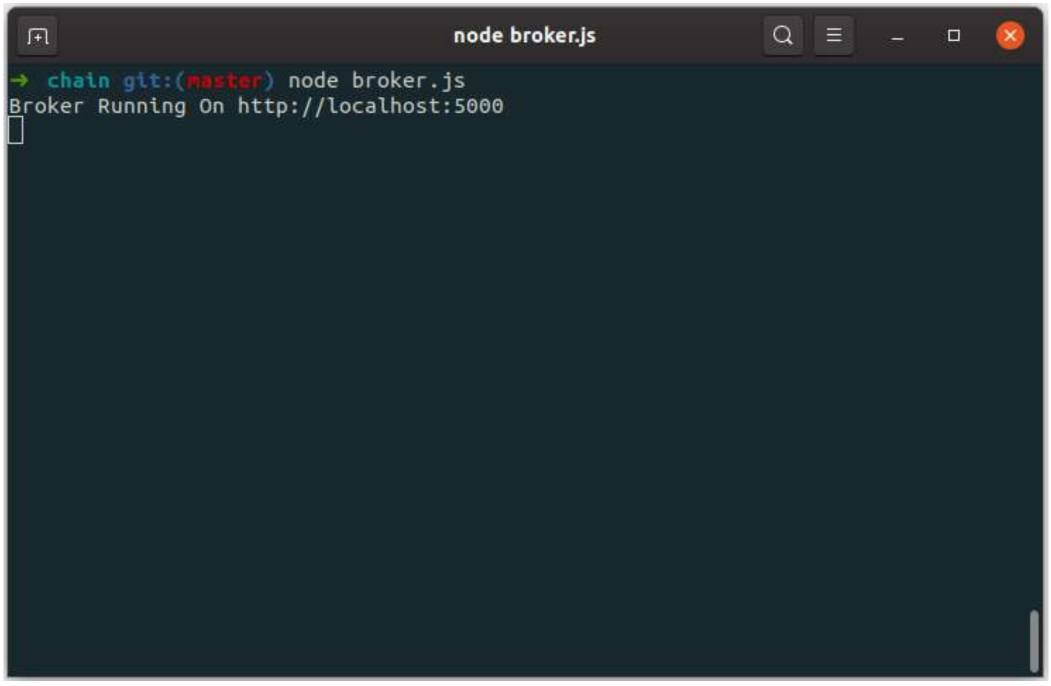
All the EVMs in the network are connected to each other in the system to form a full mesh network. Any two nodes in the network communicate with each other using a duplex bidirectional data stream, thus forming a peer to peer network. EVM nodes are able perform a number of operations on the blockchain like registering votes, creating new blocks, fetch newly created blocks from other nodes in the network and request latest instance of the blockchain.

8.2 RESULT

In this project, we have designed a security mechanism based on blockchain technology to be integrated with the EVMs in order to maintain the integrity of votes in the election process. Suitable modifications to the original implementation of the blockchain by Satoshi Nakamoto [4] has been suggested to suit the requirements of a voting system. Since the proposed design makes use of a decentralized network, when any malicious change is made to the votes stored in the blockchain then the chain tampering is noticed and the tampered block is replaced with the correct one from another randomly chosen node in the next mining cycle thus protecting the data from modifications. In order to successfully modify the data stored in the blockchain, the change needs to be reflected at the data stored in all the nodes in the network. Along with this, the data stored in all the subsequent blocks at all the nodes will also have to be changed to maintain the cryptographic link between all the blocks. In this way our model can ensure the integrity of the votes at all times.

8.2.1 Broker Initialization

Broker utility acts as the connection manager in order for the nodes to connect to each other. It is also responsible to maintain a list of currently active nodes in the system. As the implementation consists of nodes connected to each other to form a p2p structure the, it faces scalability issues when the number of nodes increases beyond a certain point. To overcome this problem the large network of independent nodes is broken down into a number of smaller mesh networks and each broker acts as a connection manager for nodes in its respective network. The broker also allows inter-network connection between the nodes from different sub-networks. The broker service only needs to be run manually for the very first time, all the next instances of the broker are created and destroyed automatically by the system as per requirements. Fig 8.2.1.1 shows the command to run the broker utility.

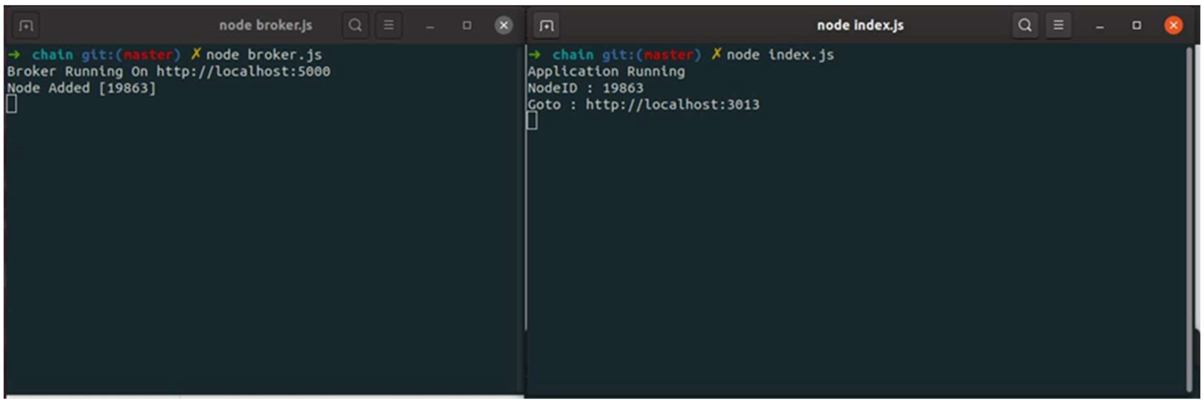


```
→ chain git:(master) node broker.js
Broker Running On http://localhost:5000
```

Fig 8.2.1.1 – Broker Initialization

8.2.2 Broker Registry

Each node entering or leaving the network must report to its respective broker that it has Entered or Disconnected from the system. This allows the remaining nodes in the sub-network to initiate / terminate connections with the entering or leaving node. This keeps the network up-to-date and reduces latency in the subnetwork by eliminating the unnecessary relay of information to the node that is no longer present in the network. Fig 8.2.2.1 shows how a new Node joining the network, announces its arrival to the broker. The broker in turn acknowledges the arrival of the new node and adds it to the list of active nodes in the sub-network.



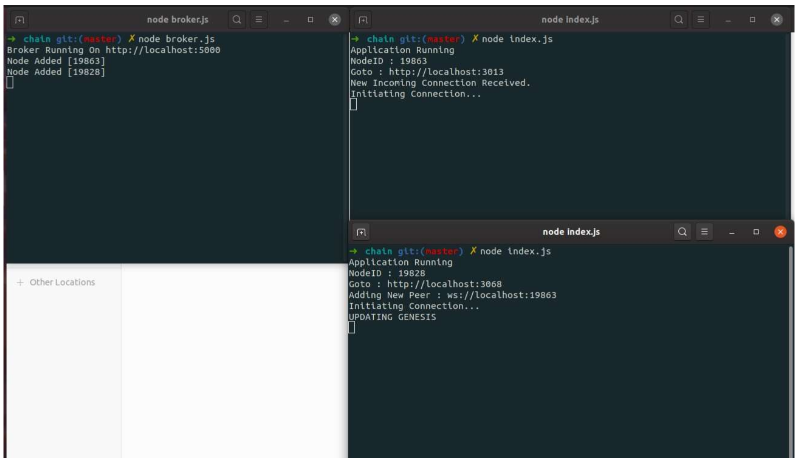
```
➜ chain git:(master) ✗ node broker.js
Broker Running On http://localhost:5000
Node Added [19863]
```

```
➜ chain git:(master) ✗ node index.js
Application Running
NodeID : 19863
Goto : http://localhost:3013
```

Fig 8.2.2.1 – Broker Registry

8.2.3 P2P Network Establishment

When a node announces its arrival to the broker service, the broker in turn provides the new incoming node with the list of already existing nodes (if any) in the sub-network. The incoming node then uses this list to initiate connections with all the active nodes in the network so as to form a mesh network. At last, the broker adds the credentials of the newly joined node to list of active nodes in the network. Fig 8.2.3.1 shows how two nodes establish connections to each other using the broker service.



```
➜ chain git:(master) ✗ node broker.js
Broker Running On http://localhost:5000
Node Added [19863]
Node Added [19828]
```

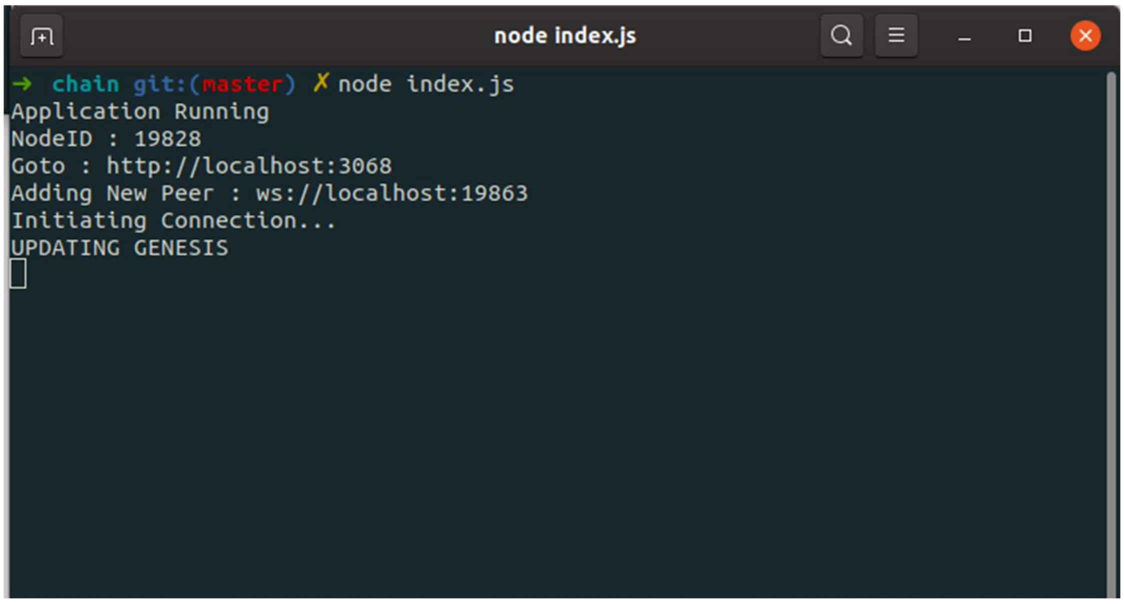
```
➜ chain git:(master) ✗ node index.js
Application Running
NodeID : 19863
Goto : http://localhost:3013
New Incoming Connection Received.
Initiating Connection...
```

```
➜ chain git:(master) ✗ node index.js
Application Running
NodeID : 19828
Goto : http://localhost:3068
Adding New Peer : ws://localhost:19863
Initiating Connection...
UPDATING GENESIS
```

Fig 8.2.3.1 – P2P Network Initialization

8.2.4 Chain Updates

Whenever a node has successfully connected to the network, the first thing it needs to do is to update its instance of the Blockchain. To do this, we have used the "Longest Valid Chain" technique. The new node requests all nodes in the network. After the nodes has received the instances of chains from all other nodes, it validates all the Chains and select the longest valid chain and sets it as its own Blockchain instance. The node is then ready to begin operations.

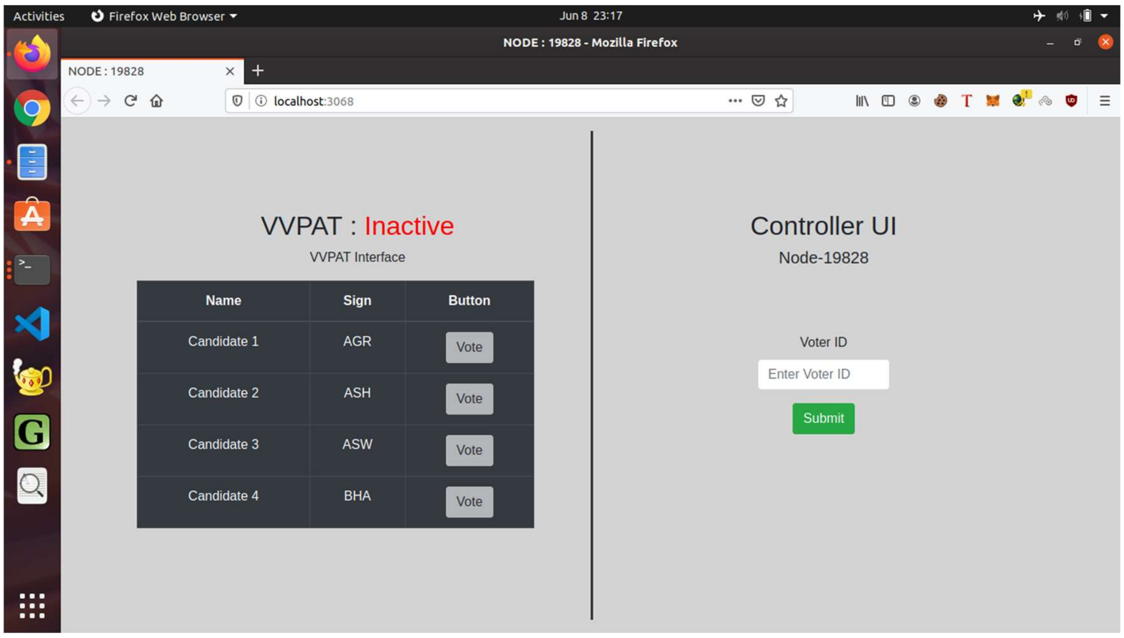


```
→ chain git:(master) ✗ node index.js
Application Running
NodeID : 19828
Goto : http://localhost:3068
Adding New Peer : ws://localhost:19863
Initiating Connection...
UPDATING GENESIS
```

Fig 8.2.4.1 – Chain Updates

8.2.5 Controller UI Form Simulation

The VVPAT is electrically interfaced with the controller's PC at the voting station and the voting processes can begin then. The controller's PC and the VVPAT works in coordination with each other. The VVPAT is in inactive state until explicitly signalled by the controller. When the voter arrives at the voting station, the person in charge at the voting station checks if the VoterID of that voter is used to previously cast vote. Once it is ensured that the VoterID has not been used before, the VVPAT is unlocked and the Voter proceeds to cast his vote using the VVPAT. Once the vote is registered, the confirmation is displayed to the voter and the VVPAT returns to the LOCKED state.



VVPAT : Inactive

VVPAT Interface

Name	Sign	Button
Candidate 1	AGR	Vote
Candidate 2	ASH	Vote
Candidate 3	ASW	Vote
Candidate 4	BHA	Vote

Controller UI

Node-19828

Voter ID

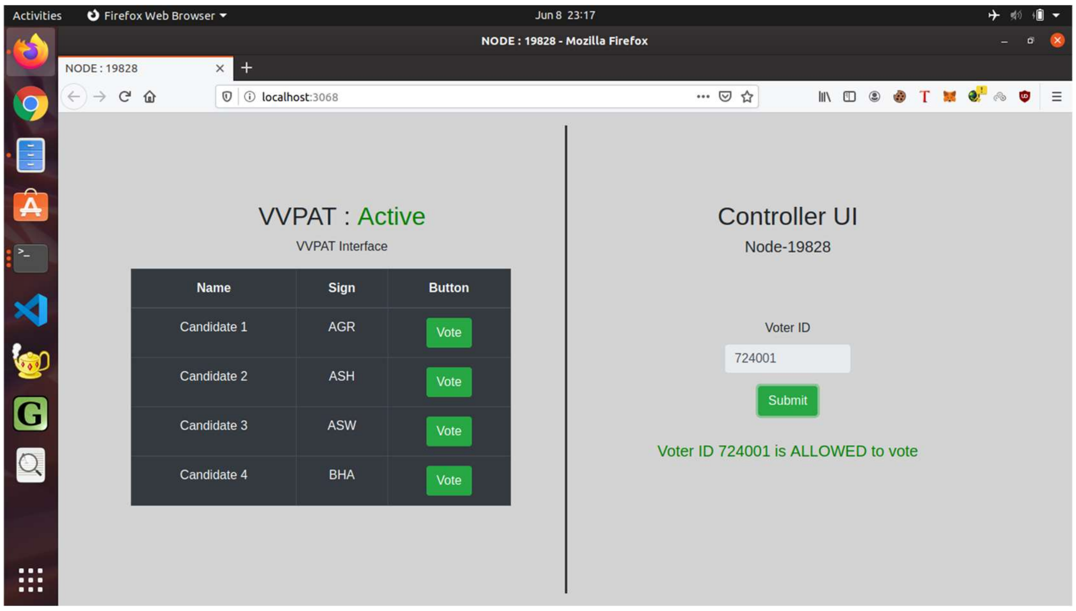
Enter Voter ID

Submit

Fig 8.2.5.1 – VVPAT In Locked State

Each EVM in the network is interfaced with a controller station which is used to switch the EVM between locked and unlocked state. The EVM cannot interact with the blockchain when it is in the locked state. When the voter arrives at the voting booth, the voter interacts with the officials present at the controller station where the identity of the voter is manually verified by the authorities. The officials then enter the voter's information at the controller station from where it is sent to the verification subsystem to fetch other information about the voter.

The verification subsystem searches the voter's databases in order to retrieve the voter's information. This information also includes the flag that indicates if the voter has already voted. This information is then sent back to the controller station where this information is manually checked again. If the status flag is false or not voted, then the controller station forwards the voter's information to the EVM and signals the EVM to switch to the unlocked state.



VVPAT : Active

VVPAT Interface

Name	Sign	Button
Candidate 1	AGR	Vote
Candidate 2	ASH	Vote
Candidate 3	ASW	Vote
Candidate 4	BHA	Vote

Controller UI

Node-19828

Voter ID

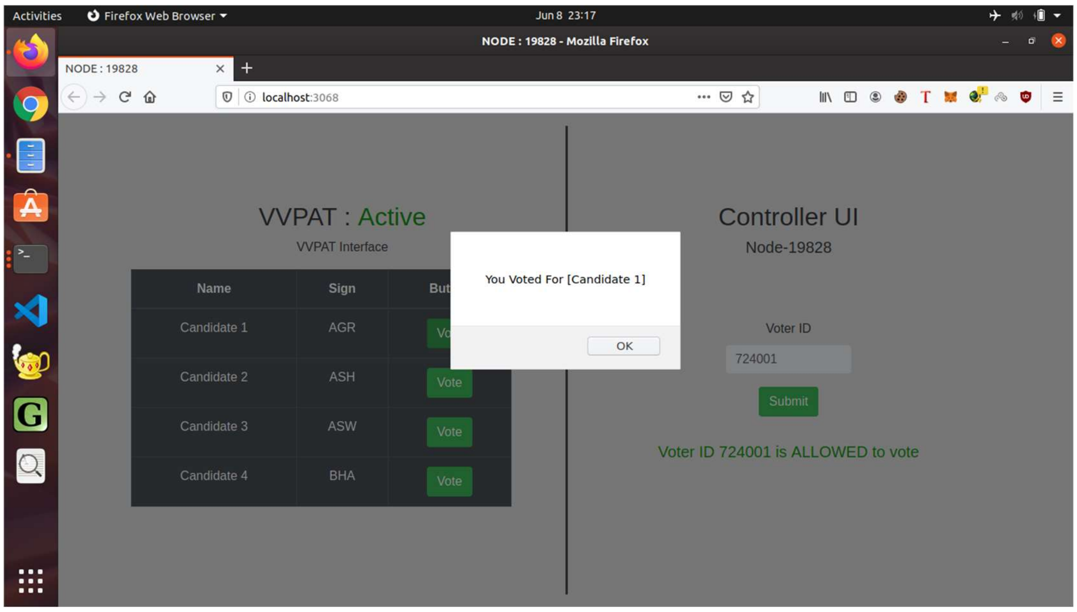
724001

Submit

Voter ID 724001 is ALLOWED to vote

Fig 8.2.5.2 – VVPAT In Unlocked State

The EVM is now ready to register vote for the provided voter-id. The voter then proceeds to the EVM and presses the button corresponding to the candidate of his choice. Upon successfully registering the voter’s input, the EVM notifies the voter that the vote is successfully cast and the EVM goes back to the locked state.



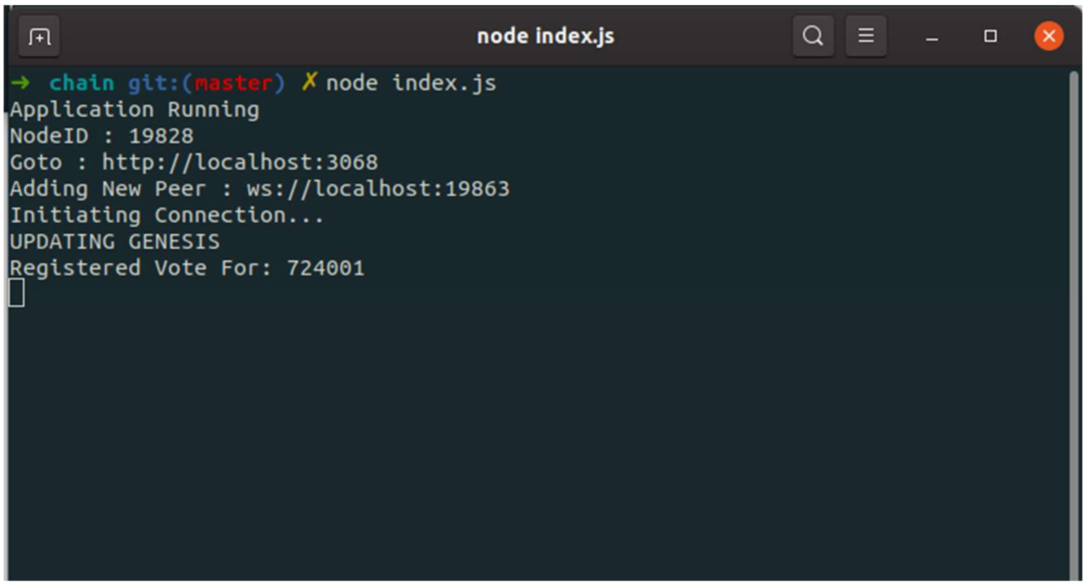
You Voted For [Candidate 1]

OK

Fig 8.2.5.3 – Vote Successfully Cast

8.2.6 Vote Registration

Once the vote is registered by the VVPAT, this is where the blockchain's operations are initiated. The registered vote is added to the pending transactions of the node. All votes registered between two mining cycles are maintained in the pending transactions list. The system uses a mining cycle of 15 minutes, i.e. a new block is created by each node every 15 minutes. Once the mining cycle is completed all the votes in the pending transactions list are added to a new block and this new block is then then sent to all other nodes in the network. When the transactions (votes) are added to the new block, the CandidateIDs in the vote transactions are encrypted in order maintain vote secrecy.

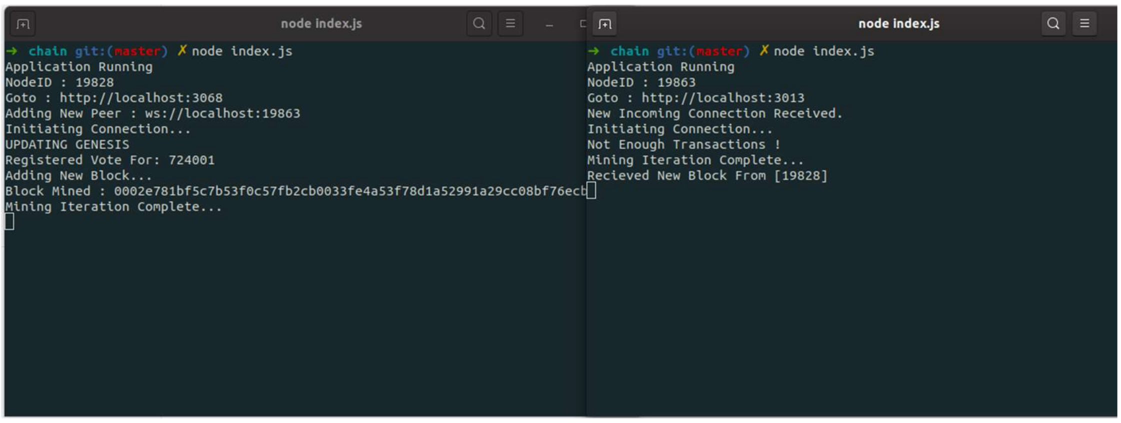


```
→ chain git:(master) ✗ node index.js
Application Running
NodeID : 19828
Goto : http://localhost:3068
Adding New Peer : ws://localhost:19863
Initiating Connection...
UPDATING GENESIS
Registered Vote For: 724001
```

Fig 8.2.6.1 – Vote Registration

8.2.7 Block Mined

When the block is mined, it is sent to all the nodes in the network. All nodes upon receiving the block check the validity of the block and its transactions. This is done to ensure that no fraudulent transactions are approved by the system and the all the nodes in the system maintain the same version of the Blockchain. If the block is found to be invalid, it is rejected and is not added to the Blockchain by the nodes. Only when the block and its containing transactions are validated by all the nodes in the network, it is then added to the Blockchain and all the nodes update their instance of the chain.

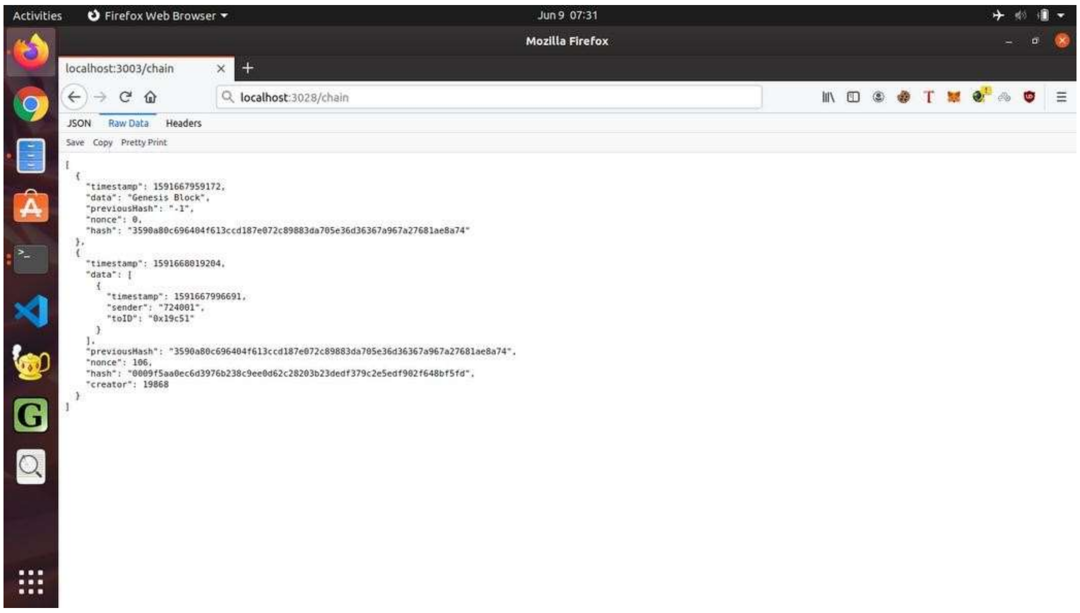


```
→ chain git:(master) ✗ node index.js
Application Running
NodeID : 19828
Goto : http://localhost:3068
Adding New Peer : ws://localhost:19863
Initiating Connection...
UPDATING GENESIS
Registered Vote For: 724001
Adding New Block...
Block Mined : 0002e781bf5c7b53f0c57fb2cb0033fe4a53f78d1a52991a29cc08bf76ecb
Mining Iteration Complete...
```

```
→ chain git:(master) ✗ node index.js
Application Running
NodeID : 19863
Goto : http://localhost:3013
New Incoming Connection Received.
Initiating Connection...
Not Enough Transactions !
Mining Iteration Complete...
Recieved New Block From [19828]
```

Fig 8.2.7.1 – Block Mined

Fig 8.2.7.1 shows the status and contents of the chain after a new block containing the votes is added to the chain.



```
[
  {
    "timestamp": 1591667959172,
    "data": "Genesis Block",
    "previousHash": "-1",
    "nonce": 0,
    "hash": "3590a80c696404f613ccd187e072c89883da705e36d36367a967a27681ae8a74"
  },
  {
    "timestamp": 1591668019204,
    "data": [
      {
        "timestamp": 1591667996691,
        "sender": "724001",
        "toID": "0x19c51"
      }
    ],
    "previousHash": "3590a80c696404f613ccd187e072c89883da705e36d36367a967a27681ae8a74",
    "nonce": 106,
    "hash": "0009f5aa0ec6d3976b238c9ee0d62c28203b23dedf379c2e5edf902f648bf5fd",
    "creator": 19868
  }
]
```

Fig 8.2.7.2 – Final Chain Status

9. CONCLUSION

The existing voting system is having the big number of problems which might lead to political distraction of a nation. Transparent balloting system plays a vital role in a democracy to regulate free and fair elections. For our design we tried to create a system that doesn't entirely replace the current voting but rather integrates within a current system. We decided to do this to allow for as many different ways to vote as possible, this is so voting can be accessed by the majority of the population. The proposed system overcomes most of the disadvantages of the traditional system by providing voter privacy, security and also transparency where the user is allowed to verify their votes. In addition to this, the system allows the users to vote from their nearby location, thus achieving maximum voting percentage. The proposed system is cost-efficient when compared to the traditional electronic voting machines. Our proposed system can be further enhanced by replacing OTP verification by fingerprint verification or face recognition in real time implementation. So that electoral fraud and voter impersonation can be eradicated to the maximum.

10. REFERENCES

1. Nicholas Weaver, “Secure the vote today,” Available at: https://www.lawfareblog.com/secure-vote-today
2. Wolchok, et al. , “Security analysis of India’s electronic voting machines,” Proceedings of the 17th ACM conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010.
3. Adrià Rodríguez-Pérez, “Secret suffrage in remote electronic voting systems”, Fourth International Conference in eDemocracy & eGovernment (ICEGEG) pp. 227-228
4. Satoshi Nakamoto, “Bitcoin: a peer-to-peer electronic cash system.” Available at: http//bitcoin.org/bitcoin.pdf
5. Friðrik Þ. Hjálmarsson, Gunnlaugur K. Hreiðarsson, Mohammad Hamdaqa, Gísli Hjálmtýsson, “Blockchain-based e-voting system,” 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, USA.
6. Yuan Yuan, Yanyan Hu, Shaohua Huang, Shengjiao Cao, Anuj Chopra, “A privacy-preserving voting protocol on blockchain”, IEEE 11th International Conference on Cloud Computing, 2018
7. Patrick McCorry, Siamak F. Shahandashti, Feng Hao, “A smart contract for boardroom voting with maximum voter privacy,” Financial Cryptography and Data Security, 2017, Volume 10322
8. Arya Sahadevan, Deepa Mathew, Jairam Mookathana, Bijoy Antony Jose, “An offline online strategy for IoT using MQTT,” 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing

9. Agora (2017), “Agora: Bringing our voting systems into the 21st century”, Available at: https://www.agora.vote
10. Sagar Shah, Qaish Kanchwala, Huaiqian Mi, “Block chain voting system,” Northeastern University, 2016
11. Mahdi H Miraz and Maaruf Ali, “Blockchain enabled enhanced IoT ecosystem security,” Int. Conf. on Emerging Technologies in Computing 2018 (iCETiC ’18), London Metropolitan University, London, UK
12. Daniel Dob, “Permissioned vs Permissionless Blockchains: Understanding the differences,” January 7, 2020. Available at: https://blockonomi.com/permissioned-vs-permissionless-blockchains
13. Sobti, Rajeev & Ganesan, Geetha, “Cryptographic Hash Functions: A review,” International Journal of Computer Science Issues (2012), Volume 9 p461-479
14. Yirendra Kumar Yadav, Soumya Batham, Mradul Jain, Shivani Sharma, “An approach to electronic voting system using UIDAI”, 2014 International Conference on Electronics and Communication Systems (ICECS-2014), Coimbatore, India, February 13-14, 2014