Electronic Voting Using Decentralized System Based on Ethereum Blockchain

Fajri Fadli, Singgih Jatmiko dan Missa Lamsani

Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Gunadarma
Jl. Margonda Raya No. 100, Depok 16424, Jawa Barat
E-mail: ajiefajrifadli@student.gunadarma.ac.id, singgih@staff.gunadarma.ac.id, missa@staff.gunadarma.ac.id

Abstrak

Pemilihan Suara Secara Elektronis Menggunakan Sistem Terdesentralisasi Berbasis Blockchain Ethereum merupakan suatu sistem yang dibuat dalam bentuk aplikasi yang dapat digunakan dalam pemilihan umum. Hal ini dilakukan untuk menyelesaikan masalah yang dimiliki oleh sistem pemilihan suara secara elektronis yang konvensional dimana integritas data belum terjamin dan data hasil suara kemungkinan dapat dirubah oleh pihak tertentu. Masalah ini dapat diselesaikan dengan membuat sistem pemilihan suara secara elektronik yang menyimpan data hasil suara pada blockchain agar integritas data dapat terjamin. Pembuatan sistem ini dilakukan menggunakan bahasa Solidity, bahasa pemrograman yang sudah Turing Complete pada Ethereum. Source code yang dijalankan akan di kompilasi menjadi Bytecode yang kemudian dijalankan pada Ethereum Virtual Machine. Program yang dibuat tersebut kemudian akan di desentralisasi kan menggunakan Blockchain. Hasil nya adalah sistem Electronic Voting yang terdesentralisasi. Sistem ini dapat diakses melalui website https://pemilurt.herokuapp.com/.

Kata Kunci: Blockchain, Electronic Voting, Ethereum, Sistem Terdesentralisasi, Smart Government.

Abstract

Electronic Voting Using Decentralised System Based On Ethereum's Blockchain is a system made in the form of application used for Electronic Voting. This is done to solve the current problem on conventional electronic voting system where the integrity of the data can't be ascertain and the result of the vote can be tampered malicious actors. To solve this problem, an electronic voting system will be built that store the data of election results on blockchain to ensure the integrity of the data. The creation of the system involves using the Solidity language, a Turing Complete programming language used on Ethereum. Source code that will be run will first need to be compiled into Bytecode, which will then in turn run on Ethereum Virtual Machine. The finished program will then be decentralised using Blockchain. The result will be a decentralised Electronic Voting system that can be accessed from the website https://pemilurt.herokuapp.com/.

Keywords: Blockchain, Decentralised System, Electronic Voting, Ethereum, Smart Government

Introduction

Voting are one of the methods used to decide choices to be made on nations that uphold democracy. Dating back to the ancient Athens, voting itself has a simple concept. People give their vote on a certain matter and then the result will be counted one by one for all people to see. Today, voting usually done as part of an election. This days however, with a large number of voters distributed over a wide location, the classical way of doing voting is terribly expensive and takes a lot of time. The time it takes to fully count the result of the vote could be exploited by malicious party who spreads hoax and fake news in order to create unrest and polarizing the voters. We could see this happen in Indonesia on it's 2019 general elections, where hoax and fake news spread through social media eventually culminates in massive demonstration. This demonstration leads to the loss of life for 8 people, damages to cars and buildings at the site where the demonstration took place, temporary restriction on social media and internet communication, as well as the falling value of Rupiah due to concern over Indonesia's political stability.

Electronic voting or E-voting seeks to remedy this problem in a number of ways. Currently there are two kind of E-voting used in the world. The first is by using Direct-Recording Electronic (DRE)

DOI : http://dx.doi.org/10.32409/jikstik.19.1.2718

voting machine. By using this machine, voters can cast their vote and the result of it will either be stored on each machine, or transmitted to a central database where it will be counted. While this could simplify tallying the voting result, the machine still need to be distributed to voting stations, thus not entirely addressing the issue currently present on manual voting or paper-based voting. Furthermore, the machine itself could be tampered by malicious actor, as proven by Blaze et al [1].

The second method of E-voting is by using internet. This method is widely used in Estonia, though it haven't found much traction on other country. The voters can cast their vote by scanning their national ID card and then use their computer to give their vote. The vote will then be transmitted through internet to a centralized server that store and tally the vote result. With no paper or machine to be distributed and the system's ability to tally the vote count, it has addressed the issue present for manual or paper-based voting. However, reports from Organization for Security and Co-operation in Europe/Office for Democratic Institutions and Human Rights (OSCE/ODIHR) has voiced some concern on the viability of this method. One of them was the concern that whether the vote cast by voter is and the vote tallied on the central server is actually hasn't been tampered. Methods such as Man-in-the-middle attack could alter the vote sent through internet, leaving the voters none the wiser. Another concern is that the central server can be attacked by malicious actor, and if successful can change the result of the vote itself [2]. Also, both DRE based voting as well as I-Voting doesn't publicly give their source code, raising concern on transparency as well.

If left unresolved, this will eventually lead to the public's loss of trust in voting and election process, as well as the legitimacy of the democratic system itself in the eyes of the people.

Thus we have turned to blockchain to address this issue. The benefits of using blockchain are immutability, transparency, security and reliability. Immutability, because the data is stored in multiple nodes and even if one of the node has it's data tampered, the data on other nodes will not be affected. In order to actually tamper with data stored on blockchain, any malicious actor will need to not only attack a single node in the blockchain network, but at least 51% of nodes connected on the blockchain network. Transparency, due to blockchain nature of open and distributed ledger, therefore the data stored on the blockchain can be available for the public. Security and reliability, for example against Denial of Services Attacks, that can be countered due to blockchain nature as a decentralized system. If any one of nodes on the blockchain network can't operate, there will be other nodes that can take the load off the inactive nodes. There are a number of papers that have explored this idea [3][4][5][6] at around the same time. This paper will contribute in explaining the implementation of blockchain, particularly using Ethereum blockchain, as a method for electronic voting for neighborhood association in Indonesia (locally known as Rukun Tetangga or RT), allowing it's voters to vote without needing to have an Ethereum account or any ethers to vote.

Blockchain

Blockchain is essentially a distributed ledger filled with records of transactions. This means that this records of transactions is not owned by just one party that need to be trusted, but every node in the blockchain network will have this ledger [7]. The content that the records store in the ledger can be anything, from it's first implementation by the pseudonym Satoshi Nakamoto to store records of digital cash now known as Bitcoin, to storing general use data such as a vote result, which this paper will explore. As it's name implies, the records or data is stored in a chain of blocks. Each block also contains the cryptographic hash of the previous block. This helps in ensuring the data stored on blockchain is immutable, as any changes made to one block will change it's cryptographic hash that the next block refers to. Any discrepancy of data will be compared with data from other nodes. If the majority of other nodes doesn't have that same discrepancy, then the changed data will not be verified. Thus ensuring the integrity of the data [8].

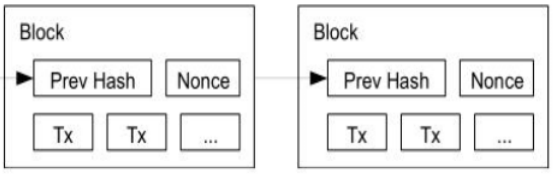

Figure 1: Illustration of a Blockchain Model

Ethereum

Ethereum was first developed to bring blockchain from only storing records of financial transaction as Bitcoin did, but to also store a more general data by bringing the concept of Smart Contract. Smart Contract is similar to a class in object-oriented programming, where there could be multiple instances from one Smart Contract. This instances will then be executed on top of Ethereum Virtual Machine (EVM) by the nodes on Ethereum blockchain. Each node that works on the Smart Contract will be incentivised by given a small amount of ether, the cryptocurrency that Ethereum uses. This allows development of a decentralized system that can be used for all manner of things, not just for digital cash [9]. The programming language Solidity was made in order to facilitate development of decentralized system on top of Ethereum. While there

are other programming language that can be used to create Smart Contract, Solidity has similar syntax compared to existing programming language like Javascript, Java, and C, which make Solidity popular for development of Smart Contract [10].

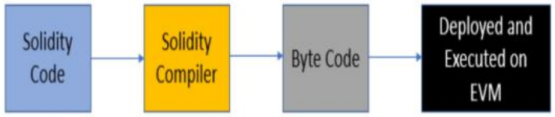

Figure 2: Workflow on how Solidity code deployed and executed on EVM

To use Ethereum blockchain, we will need an Ethereum wallet that can store some amount of ether to be paid for the execution of our smart contract. To that end, we will use Metamask. Metamask is an Ethereum wallet that can be accessed from the web browser. You can use Metamask for storing ether and sending ether to another Ethereum account [11].

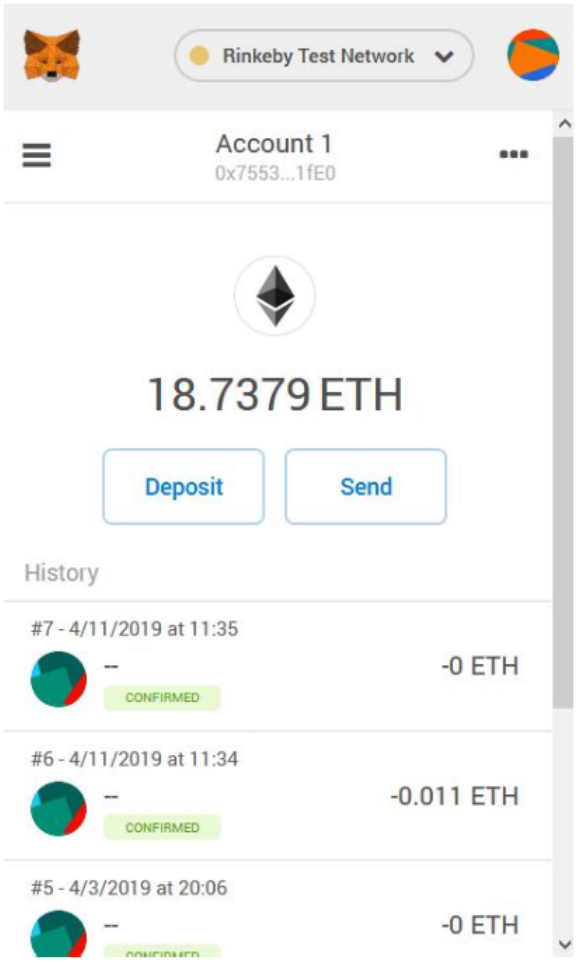

Figure 3: Metamask User Interface

Electronic Voting System using Ethereum Blockchain

The electronic voting system that we have made is divided into 3 main parts. First is the front-end, which will be how the user interact with the system. The front-end is a website that the user can access from their web browser on their electronic device. The second part is the back-end, which will store user authentication data. This way the user doesn't need to have an ethereum account or any ether to be able to vote. The third part is the blockchain, which will be the creation of a smart contract that allows users to vote on their chosen candidate as well as election organizer to register any new candidates for the next election.

Table 1: Parts of system and it's explanation

Parts	Software	Explanation
Front-end	Javascript, JSX, ReactJS, Bootstrap	The front-end will be the website that the users can access in order to give their vote
Back-end	MongoDB , NodeJS	The back-end will be the database and API that will authenticate the users from data based on the users national ID card
Blockchain	Ethereum, Solidity, Ganache, Metamask ,Infura, Web3	The blockchain will be the smart contract that will then be deployed and keep track of candidates and their votes

The interface for the systems will be divided in accordance with the navigation structure that can be seen in Figure 4.

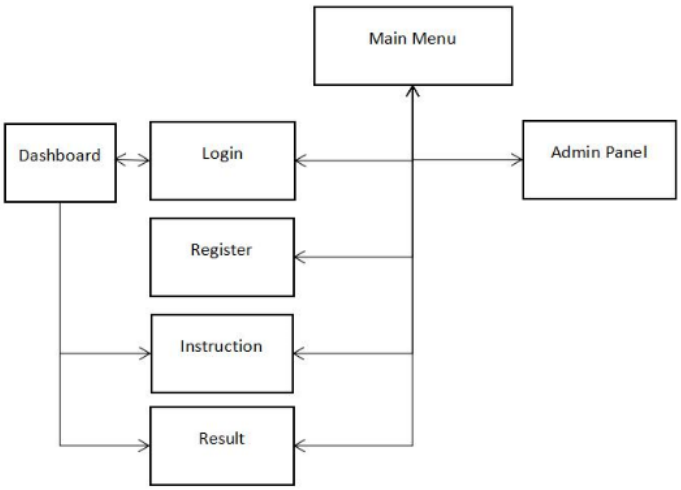

Figure 4: Navigation Structure

As we can see from Figure 4, from the Main Menu, users will be able to access the Login, Register, Instruction, and Result page. The content of

the result itself will only available after the voting period has ended. In order to access the Dashboard where the user can vote, they will need to register from the Register page. After registration, they can access the Login page, enter their credential, then access the Dashboard to vote. The Admin Panel will only be available by login through Metamask, one of the ways to get an Ethereum account. If the logged account address is the same as the account address specified in the system, only then will the Admin Panel can be accessed.

There will be two types of users, voters and admin. Voters will be able to check who they want to vote, their vision and mission, and finally to vote for their chosen candidate. Admin will be able to open and close the period of voting, register candidates, and reset the result of past voting.

The system itself is divided into numerous class that works on different parts of the system, which could be seen on the diagram in Figure 6. We can see from the diagram at Figure 6 that the system has been divided into 6 classes. This classes has their own method that ensure that the system will be usable by it's users.

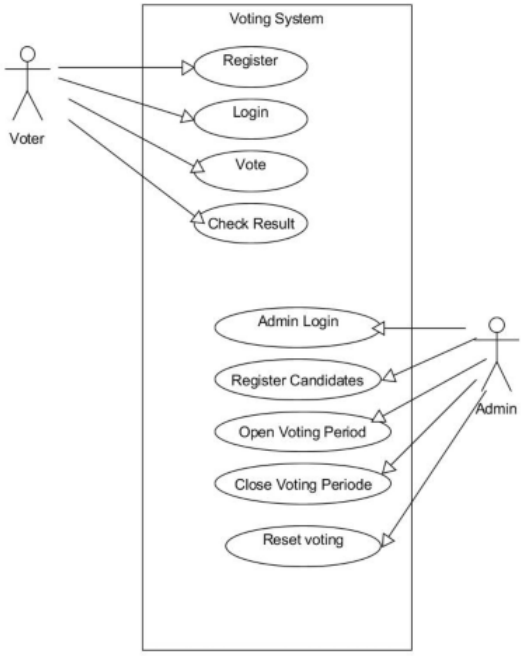

Figure 5: Use Case Diagram

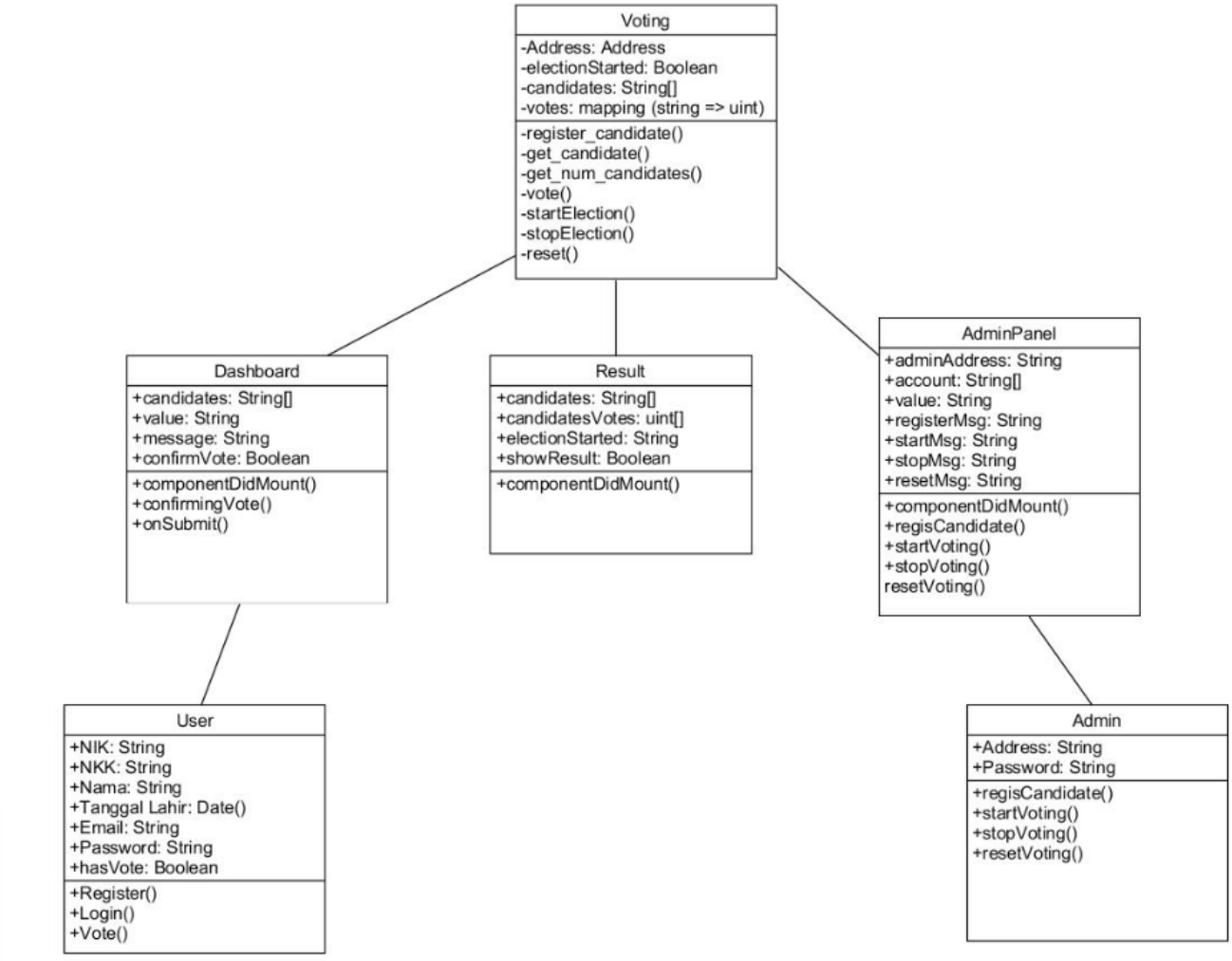

Figure 6: Class Diagram

To see how each class communicate to fulfill a specific use case, we can refer to the sequence diagram in Figure 7.

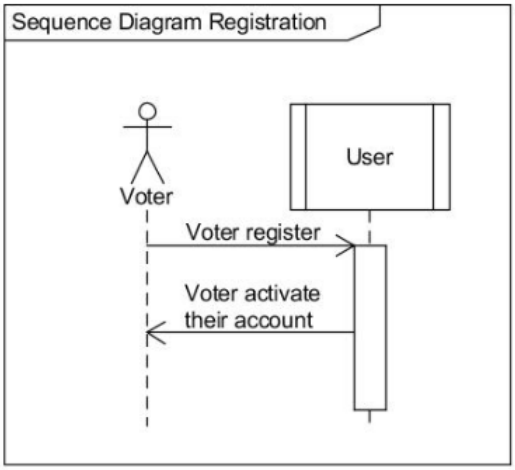

Figure 7: Sequence Diagram for Registration

Here we can see that voter will need to register to have an account that will be used for voting. This is handled by the User class. Then they will need to activate their account from their email address. Once that is finished, voter can login to their account, which is displayed in the sequence diagram in Figure 8.

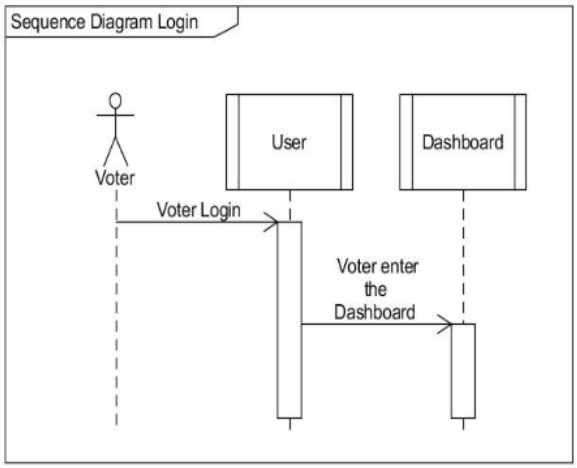

Figure 8: Sequence Diagram for Login

From the diagram in Figure 8, we can see that voter will login which will be handled by the User class. Then the voter can access the dashboard to vote on their chosen candidate which will be handled by the Dashboard class.

Then voter can choose their chosen candidate to vote for which will be handled by the Dashboard class. The Voting class, which is the smart contract that we have built, will store the vote for the chosen candidate on the blockchain.

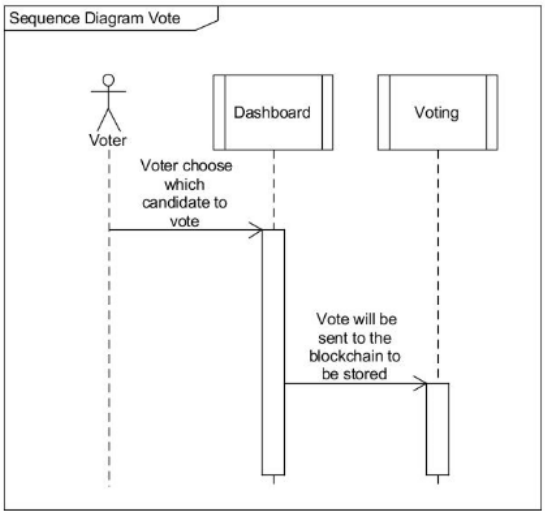

Figure 9: Sequence Diagram for Voting

Development Process of Electronic Voting System using Ethereum Blockchain

On developing this system, we use Version Control System (VCS) to provide control and manage changes on the source code. This allows us to track the development progress made on the system by looking at it's commit history, as well as reverting unwanted changes to the source code. We will use Git as our VCS. Originally created by Linus Torvalds for development of the Linux Kernel, nowadays it's used for all manner of software development projects. We will also use a feature of Git that allows us to upload our managed source code to a code repository. This code repository can be seen by the public, hence solving the issue of transparency that still plague conventional electronic voting system. As can be seen at Table 1, this system is divided into three parts. The development of the front-end will focus on creating a user friendly web application that can be used from either computer or mobile phone. The development of the back-end will focus on creating API that authenticate users based on data from national ID card. The development of the blockchain will involve the creation of smart contract which will function as the overall logic of the voting system, which will be then deployed to the Ethreum blockchain network. The code from all three parts will be managed using Git. It will then be uploaded to code repository that can be accessed by the public. The code repository for this system can be accessed at https://github.com/FajriFadli/voting-ethereum

Testing

In order to test that the smart contract has been created with no flaw, we will need to test it. We will

use an automated testing method called unit testing that allows to test each function on the smart contract and whether it works as expected or not. We will use Mocha, a unit testing framework to test it. There are seven aspect that will be tested, which is:

1. Is the smart contract successfully deployed on the Ethereum blockchain?
2. Is there any authorized Ethereum address that will be used for voting?
3. Is the smart contract can be used to register candidates and check the number of candidates?
4. Is the smart contract can be used to vote when the voting period has open?
5. Is the voting can only be done from authorized Ethereum address?
6. Is the voting can only be done on voting period?
7. Is the value from past voting will be turned to zero when it's reset?

We want all the answer to this question to return as true. To answer this question, we will create a test script that return a value that will then be compared to an expected value. If the returned value match or equal the expected value, then the answer will be true and the aspect that is tested is completed without flaw. If it's not, then the answer will be false and the aspect that is tested has some flaw that will need to be fixed before the smart contract deployed to the blockchain. An example of the test can be seen in Table 2.

Table 2: Mocha Script for Unit Testing Test case Script Explanation

Test case	Script	Explanation
Is the smart contract successfully deployed on the Ethereum blockchain?	it("Contract has been deployed", () => { assert.ok(voting.options.address); });	It checked whether the smart contract has been deployed by asserting that it has an ethereum address. The test case fails if the smart contract does not has an ethereum address

From testing with Mocha, we can see the testing result in Figure 10.

From the test result in Figure 10, we can conclude that all seven aspect has passed testing, which we can turn into the data in Table 3.

The data at Table 3 shows that all the tests has been passed. Each test case has it's own script that works by asserting that the specified test case has been met, an example of which can be seen at Table 2. The script used as part of the testing will be uploaded to a public code repository at https://github.com/FajriFadli/voting-ethereum/tree/master/blockchain/test . This will ensure that the public can audit and verify this test on their own, if they choose to do so.

Figure 10: Testing Result using Mocha

Table 3: Test Result Test case Test Result

Test case	Test Result
Is the smart contract successfully deployed on the Ethereum blockchain?	Passed
Is there any authorized Ethereum address that will be used for voting?	Passed
Is the smart contract can be used to register candidates and check the number of candidates?	Passed
Is the smart contract can be used to vote when the voting period has open?	Passed
Is the voting can only be done from authorized Ethereum address?	Passed
Is the voting can only be done on voting period?	Passed
Is the value from past voting will be turned to zero when it's reset?	Passed

Result

We have achieved a decentralized electronic voting system built on top of Ethereum blockchain that enable the voting result to be immutable so it cannot be tampered by any malicious actor while making it easy to use for the voter by not requiring the voter to have an Ethereum account or have any ether to vote. Figure 11 to Figure 18 are screen capture of the look and feel of the finished system.

And thus we have built a decentralized electronic voting system using the Ethereum blockchain which can be accessed from it's users own electronic device. It can be accessed at https://pemilurt.herokuapp.com/ . With this, it will eliminate the needs to distribute voting papers to each voting stations that can be spread over a wide area. The system that we have build is also open source, where each parts of the system can be audited and verified by the public, which in turn can open a layer of transparency for the voters and can be accessed at https://github.com/FajriFadli/voting-ethereum

Figure 11: Main Menu

Figure 12: Voter Registration

Figure 13: Account Activation

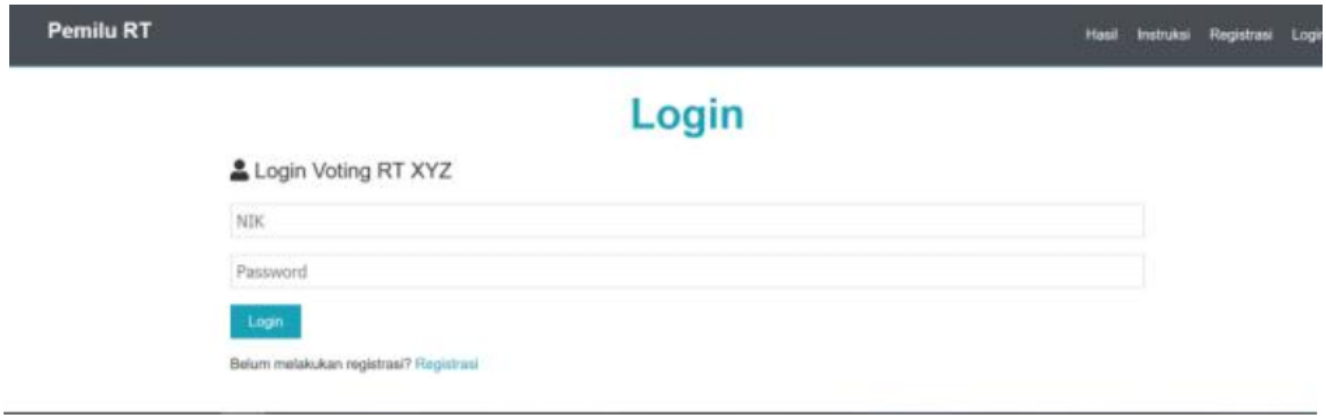

Figure 14: Voter Login

Figure 15: Voter Dashboard

Figure 16: Voter Dashboard After Choosing Desired Candidate

Figure 17: Vote Result

Figure 18: Admin Panel

Conclusion

The electronic voting system built on top of Ethereum blockchain has solved the two issues present on current method of voting, both manual or paper-based voting and conventional electronic voting, which is allowing voters to vote from their own electronic device, where voters can access the system at https://pemilurt.herokuapp.com/, and thus eliminating the issue of distribution over voting station on a wide area. It also solve the issue of transparency, where the vote result can be publically seen due to the nature of Ethereum blockchain, as well as immutable to any tampering from malicious actor. The underlying system is also open source, and can be audited and verified by the public. The code repository for this system will be available at https://github.com/FajriFadli/voting-ethereum. The system also doesn't need it's voter to have an Ethereum account or any ether at all, instead the voter use their national ID data to be able to vote. However, it's ease of use by allowing voter to use their national ID data to vote can be a double-edged sword. Social engineering that trick the voter by requesting data on their national ID could potentially use it for malicious purpose. The visual nature of the system could also hamper people with disability to vote effectively. Further integration with Accessible Rich Internet Applications (ARIA) could aid people with disability to use this system more effectively.

References

[1] M. Blaze, J. Braun, H. Hursti, D. Jefferson, M. MacAlpine, J. Moss., "Report on Cyber Vulnerabilities in U.S. Election Equipment, Databases, and Infrastructure", https://defcon.org/images/defcon26/DEFCON%2026%20voting%20village%20report.pdf , 25 September 2018.

[2] J. Lenarčič, "Estonia Parliamentary Elections OSCE/ODIHR Election Assessment Mission Report", Warsaw: ODIHR, 2011.

[3] F.S. Hardwick, A. Gioulis, R.N. Akram, K. Markantonakis, "E-Voting with Blockchain: An E-Voting Protocol with Decentralisation and Voter Privacy", https://arxiv.org/abs/1805.10258, 3 July 2018.

[4] H.V. Patil, K.G. Rathi, M.V. Tribhuwan, "A Study on Decentralized E-Voting System Using Blockchain Technology", International Research Journal of Engineering and Technology, Fast Track Publication, p48 - p53, Chennai, IRJ Publications, 2018.

[5] G.G. Dagher, B.P. Marella., M. Milojkovic, J. Mohler, "BroncoVote: Secure Voting System using Ethereum's Blockchain", Proceedings of the 4th International Conference on Information Systems Security and Privacy, (ICISSP 2018), p96-p107, Madeira, Institute for Sys-

tems and Technologies of Information, Control and Communication, 2018.

[6] A.K. Koc, U.C. Cabuk, E. Yavus, G Dalkihc, "Towards Secure E-Voting Using Ethereum Blockchain". International Symposium on Digital Forensic and Security, (ISDFS 2018), Volume: 6, Antalya, 2018.

[7] R. Modi, "Solidity Programming Essentials", Birmingham: Packt Publishing, 2018.

[8] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", http://bitcoin.org/bitcoin.pdf, 31 October 2018.

[9] V. Buterin, "A Next Generation Smart Contract & Decentralized Application Platform", https://blockchainlab.com/pdf/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf, 24 January 2014.

[10] D. Mohanty, "Ethereum for Architects and Developers", California: Apress Media LLC, 2018