EveryOne DAO

EECE 571G

Blockchain Software Engineering

The University of British Columbia

Shurui Feng

Mengmeng Li

Megan Ma

Taicheng Li

Instructor: Dr. Zehua Wang

Key words: Decentralized Autonomous Organization (DAO), Voting, Blockchain, Governance, Transparency

Table of Contents

1. Introduction

Voting provides a standard means by which individuals make democratic decisions and share their opinions on controversial topics and debates. From the dawn of democratically electing candidates, the legal system has supported the mechanism of paper ballot voting. The voters submit their identification documents to a third party, such as the organizers or administrators. After their identities are authenticated, voters mark their choices on the ballot papers to elect their representatives. The papers will then be posted in sealed envelopes and deposited in the ballot box until the election is over to count the votes (Yacoubi, 2021). Unfortunately, poll efficiency becomes the primary concern as all activities are done manually including voter verification, the casting of ballots, and ballot tabulation. The need to travel to a poll station during specified hours and wait in the lines also poses a significant challenge to voter convenience, thus affecting voter turnout (Online Voting: A Path Forward for Federal Elections, 2017).

1.1 Electronic Voting

Over the course of time, the technical advances have led to the question of whether adopting digital technology to the election process can deliver the same functionalities. With this mindset, electronic voting has evolved as a reform aimed at minimizing organizational costs due to decreasing voting overhead while simultaneously fulfilling the protection conditions of an election. E-voting systems promise to deliver numerous advantages such as enhanced voter privacy, improved accessibility, reduced environmental impact, acceleration of results processing, increases in voter turnout, and decreases in ballot errors (Online Voting: A Path Forward for Federal Elections, 2017). Electronic voting guarantees the principle of accessible voting by eliminating the need to prepare ballot papers and open polling stations, which means voters can vote from anywhere at any time as long as a suitable internet connectivity is in place (Jafar, 2021). The advent of online voting can also boost voter participation and rekindle interest in the voting system, thereby increasing voter turnover. As such, the idea of adapting digital

technology to make the voting process cheaper, quicker, and easier, can be a compelling enhancement in terms of voting administration and social issues.

1.2 Motivation

Despite all possible benefits, electronic voting technologies are viewed and used with a great deal of caution due to their potential drawbacks. Most existing online voting platforms are based on the client-server architecture where the voters must trust the third-party organizing authority for the ownership and integrity of the votes. As such, the effectiveness of online voting is determined primarily by the level of faith that voters have in the voting process (Puneet, 2021). While this topology may seem sensible for certain applications, it introduces several threats and trust concerns.

First of all, conventional electronic voting platforms lack promising measures for data integrity preservation and security protection. As voting procedures are controlled, measured, and monitored through the internet, cyberattacks targeting any combinations of undisclosed software vulnerabilities, misconfigurations, or human errors would allow a remote hacker to obtain and manipulate voting data (Online Voting: A Path Forward for Federal Elections, 2017). In addition, due to the process being centralized and licensed by the organizing authority, fallacious voting results are a possibility as a result of the central authority's dishonesty. Furthermore, a single vulnerability can lead to large-scale manipulations of votes. If the voting system is compromised, all cast votes may be at the risk of being modified or misused, thus sabotaging the poll (Puneet, 2021). Such security vulnerabilities and possible breaches which are not found with in-person, hand-counted paper voting run the danger of limiting fundamental fairness in the voting process and negatively affecting voters' trust in electrical voting.

In essence, developing an advanced digital voting platform warrants careful technical considerations to transform the online voting landscape. In regard to this, our motivation was to develop a legitimate, accurate, safe, and convenient replacement for the traditional electronic voting solutions with distributed and security protection characteristics.

2. Why Blockchain Is a Good Fit

Blockchain is a peer-to-peer distributed ledger technology for establishing trust and consensus in decentralized networks (Yacoubi, 2021). A blockchain is a data structure that is tamper-resistant in which data is organized into a chain and exchanged across a network. The network's node servers are synchronized, saving the same information across the entire network. Blockchain employs sophisticated and computationally intensive secure hash algorithms that accomplish data integrity by preventing the deletion or manipulation of data (Yacoubi, 2021). As such, it provides a secure, reliable, and decentralized system.

The Ethereum Blockchain is an open-source distributed computing platform featuring a Turing-complete scripting language where people can deploy decentralized applications (DApps) and benefit from the distribution property inherited from the Blockchain technology (Puneet, 2021). Because of several critical characteristics of its architecture which are detailed below, the Ethereum blockchain provides a viable solution to overcome the technical barriers introduced by electronic voting such as authentication, verification, ballot secrecy and auditability.

2.1 Security and Transparency

Blockchain technology is believed to have the potential of revolutionizing online voting primarily because votes are recorded and maintained on a global network of computers that is decentralized and not affiliated with any authorities (Online Voting: A Path Forward for Federal Elections, 2017). In comparison to paper-based voting or conventional electronic voting which are centralized and clogged with intermediaries, blockchain-based systems are trustless and independent, and are capable of resolving the challenge of single points of failure. Additionally, blockchain enables irreversible voting records and prevents meddling with the integrity of previous votes using a computationally expensive consensus mechanism called Proof-of-work (Yacoubi, 2021). A majority of the network nodes must reach a consensus before a new transaction becomes a permanent part of the ledger. Applied to voting, this means that votes once cast are permanently recorded and synchronized in a blockchain-wide ballot ledger. Anyone with

malicious intents would have to gain control of more than 50% of the total computing power to have the ability to change the voting history. Therefore, it would be nearly impossible for hackers to modify or delete votes under any circumstances, thus establishing a more secure and reliable voting environment.

Blockchain also ensures that the optimum level of transparency and clarity is maintained because it enables public visualization of the smart contracts and the formation of tamper-proof traces of voting. All interested parties instead of just dedicated institutions and officials can check the outcome of non-blind polls in real time using a public blockchain ledger (Yacoubi, 2021). This feature serves as an additional safeguard for fraud detection, thereby facilitating the robustness and reliability of the voting system.

2.2 User Verification

Blockchain offers a very easy means for voter verification. Normally, to take part in an online poll, voters need to identify themselves using a recognized identification system. This identification system ought to be both trusted and secure so that a voter's account cannot be stolen or used by an intruder. Building such an identification system is a complex task in itself. However, with blockchain, participants can be easily authenticated through their names and unique digital wallet addresses without a third-party server, which ensures that only authenticated users in a community are able to cast votes.

2.3 Voter Anonymity

Blockchain can easily support partial voter anonymity because it only matches voters one-to-one with a digital wallet address and records the address and voting details as a transaction history in the block. There is no direct correlation or interaction between the votes casted by the voters and their identities throughout the polling process. If a completely anonymous voting is required, a one-time ring signature technique can be employed (Wang, 2018).

To sum up, blockchains are a useful augmentation to conventional electrical voting systems due to its: (1) voting data immutability and transparency of the voting environment, (2) decentralized registration and validation mechanisms of voters, (3) unlinkable voters' identities and their votes. In view of these characteristics, blockchain has the potential to contribute toward creating a more reliable and secure solution for electronic voting.

3. Key Design Principles

During our project proposal, we gathered the blockchain advantages and possible tech overhead against the current centralized server architecture. As blockchains support those advanced features, we defined our project to be built on top of those four main principles: eligibility, transparency and verifiability, usability, and decentralization. We will explain the following feature in detail in the following paragraphs.

3.1 Eligibility

Our application only allows registered voters to vote. We have implemented some functions in the backend to verify users. Specifically, we used a modifier for the registerVote function that will check the users' previllegs. In order for users to participate in a voting event, a qualified user will be required to pay a small amount of gas fee so that we can verify and put the users' information on a chain. Since all blockchain applications require users to pay gas fees to connect to the blockchain, we also provide a website for users to get more ether if needed. In the future, we could potentially release our own token so that we can distribute our token to desiginatied users and verify the voters' eligibility on another level. The reason that we did not consider this into our current implementation is that it will waste too many resources for each event, but if we consider the scope of a US election, all costs can be well balanced.

3.2 Transparency and Verifiability

When it comes to planning a voting event or election, a key goal is to provide transparency that establishes trust and confidence during the process. Our users are able to check the results anytime unless it's blind vote. Users will be notified of the blindvote result until the voting has ended . As the website is managed by our smart contract, users can fully participate in the democratic process with the comfort of knowing the election is being managed in a transparent and secure manner. Also, since this is a public blockchain, some super users can verify the truthfulness by fetching information from the blockchain and using Web3 functions to test. I have provided some test examples in our test folder.

3.3 Flexibility and Usability

Our program also provides users with flexibility and usability with various features. as it can support tie and allows users to revote. Tie happens a lot in voting, especially in a very small sample. There are a couple ways to resolve a tie like adding more time for more users to participate. Moreover, in order to help users to save some ether, we provide the option to revote without paying an extra gas fee. As long as the event is not finished, our users can change this input at any point without paying a gas fee. In addition, there is the blind vote option which prevents users from being influenced by others' choices that may cause bias. The main reason behind this is: if we imagine this as a iclicker question, if students can see other student's vote, it will give an unfair advantage to those who vote later. Hence, we give the event creator the option to make a vote blind if others' votes are essentially influencing. We also allow users to add tags when creating an event. Later, other users apply filters that help them break down a long list of events into manageable results, which could improve user experience. Those filters include (Dao related, created by owner, and several others).

3.4 Decentralization

Unlike the traditional model of a server and user, our blockchain application is not controlled by any means by the server “killing a process or thread.” We assign each voting event with time and status variables in solidity, so that the poll events will stop by themselves, not from the event creator manually. If an event eventually ends, it will broadcast to all the users who are participating, reaching the goal of decentralization.

4. Usage scenario

We aim to provide a foundation for any community that needs decentralized governance. They can simply add an NFT-gate into our contract, and get a DAO of their community.

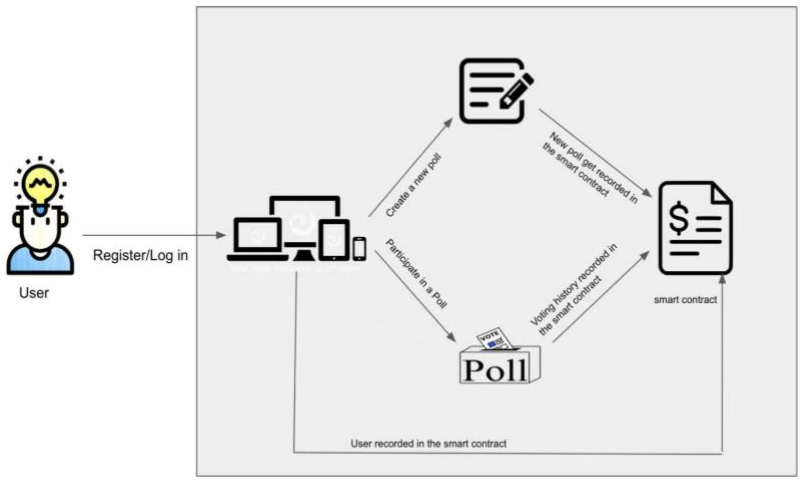

Figure 1. Usage Scenario

5. Backend implementation

In alignment with the aforementioned technical considerations and design principles, we have designed our smart contract in solidity. This section specifics the details of our backend design.

5.1 Data Structure

The following data structures are implemented in our solidity contract.

(1) State: An enum is used to represent the state of a poll, either still in voting or has already ended.

```
enum State { VOTING, ENDED }
```

(2) Selection: An enum is used to represent a maximum of eight possible selections of a poll.

```
enum Selection { DEFAULT, A, B, C, D, E, F, G, H }
```

(3) Participants: A struct data type is used to represent a record of a participant, which contains participant's Ethereum account address, name, the IDs corresponding to the polls created by the participant, and the IDs corresponding to the polls in which participants have cast votes.
A mapping type named participants is used to store the associations between Ethereum addresses and Participants.

```
struct Participant{
    address participantAddr;
    string voterName;
    uint[] pollIds;
    uint[] votedPollIds;
    }
    mapping(address => Participant) public participants;
```

(4) Participant Name: A mapping type is used to store the associations between participants' names and Ethereum addresses.

```
    mapping(string => address) public participantName;
```

(5) Polls: A struct data type is used to represent a poll created by a participant, which contains the poll's current state, ID, name, description, start time, duration, available selections, selections' descriptions, the number of total votes cast, the Ethereum addresses of those who have cast votes, the Ethereum address of the poll organizer, whether or not the poll is blind, and whether or not the poll is about DAO.

A mapping type is used to store the associations between polls' IDs and the poll events themselves.

```
struct PollEvent{
    State state;
    uint pollId;
    address organizer;
    string name;
    string description;
    uint startTime;
    uint votingDuration;
    bool blind;
    bool aboutDAO;
    Selection[] choseFrom;
    string[] optionDesc;
    uint totalVote;
    address[] voted;
}
mapping(uint => PollEvent) public polls;
```

(6) Poll Results: A struct data type is used to store the results of a poll, which consists of the current state of the poll, the poll ID, whether there is a tie in the result, all votes that have been cast, and the temporary or final result of the poll depending on the current state of the poll.
A mapping is used to store the associations between poll IDs and their results.

```
struct PollResult{
    State state;
    uint pollId;
    bool tie;
    Selection[] votedChoices;
    Selection[] result;
}
mapping(uint => PollResult) private pollResults;
```

5.2 APIs

(1) Register Participant: This works as a function to make sure users are registered or logged in to the system, and no duplicate name and addresse pair would be stored in the storage.

(2) Create Poll: Every user is able to use the API to create a versatile poll, including poll description, voting duration, some tags (blind or about DAP), and the number of choices (up to eight options could be set).

(3) Vote: Every user is able to vote on any given call by paying a small amount of gas fee, and re-vote the poll before it ends. To achieve the autonomous stop of a poll, the status of the poll will be updated (it might change from VOTING to ENDED), depending on the current timestamp the user plans to vote and the initial poll end time. **This is using similar logic to how AAVE updates its interest rate in the lending pool.** The user can only successfully vote if the updated status is still VOTING.

(4) View Result: Every user is able to check the results of a vote by paying a small amount of gas fee. If the poll is not blind, users can view the results in real-time before and after the poll has ended; if the poll is blind, users can only view the results of the poll after it has ended. It also supports the case when the result is a tie. To make sure blind poll results could not be viewed

before it ends, results viewing API would trigger a poll status update. Whenever the user is viewing results, it will update the status of a poll. Only when the poll's status is set to ENDED, a blind poll results would be displayed to users.

(5) View Poll: View function to help users view the details of a poll except for its result.

(6) Check Voted Choice: View function to help users to check whether he or she has voted before and loop up previously voted choice so that users can re-vote to the same poll easily.

5.3 Testing Using Hardhat

We tested our smart contract comprehensively using the Hardhat network, covering cases from participant registration and new poll creation to non-blind and blind poll voting. Transactions, where prerequisites were not met, have successfully come back as reverted. For casting votes and checking the outcomes, we ensured that edge cases were covered and our time-dependent logic was tested. A total of 28 test cases were designed. Below is a screenshot of our test results.

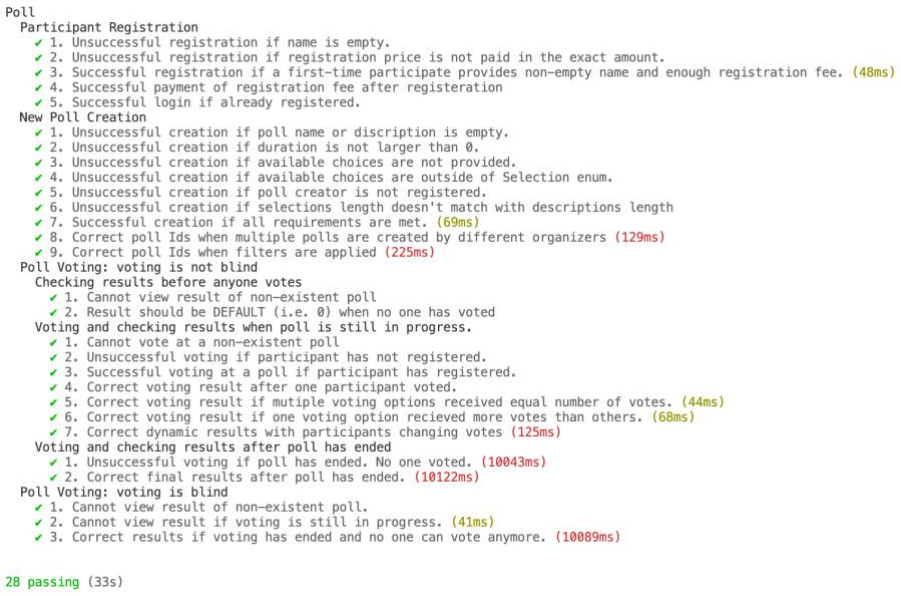

Figure 2. Test Results

6. Frontend implementation

To develop the front end, we first worked on designing an original wireframe. After analyzing the user use case, we created the wireframe and organized the front-end pages mainly into four parts: welcome page, poll dashboard, poll creation, and poll view. Drawn with Adobe XD, the wireframe is the prototype of our UX design. All the UX and UI designs are built based on the wireframe. Javascript React and React Hooks are the main techniques applied in front-end UI development. Besides, the Web3 library is also applied to interact with the Ethereum blockchain and our smart contract.

6.1 Welcome Page

On the first page of our app, users are required to connect to their Ethereum account through the Meta Mask wallet. Only when users connected with their Ethereum account and put in their user name would they be permitted to continue. The user's Ethereum account address is stored in the browser's local storage. They cannot browse the other area of our web app without this account address. On their first logged-in, they would be signed up to the system automatically. After successfully logging in, they would be turned into the Dashboard page.

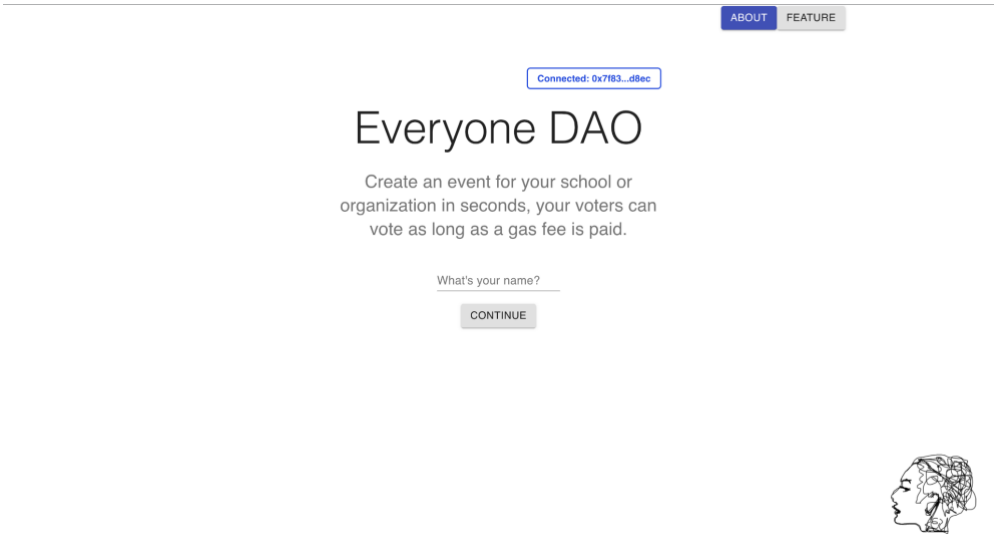

Figure 3. Welcome Page

6.2 Dashboard

All previously created polling events in our smart contract will be rendered on the dashboard. On this page, users could filter out events by the event type or by the creators to customize their events'' overview. The name, number of participants, and a short description are displayed for each poll.

On the dashboard page, users can participate in the polling event they are interested in by clicking on the PARTICIPATE button. Or view one poll's results directly by clicking on VIEW RESULTS. And also, on top of the page, users could choose to create a new polling event.

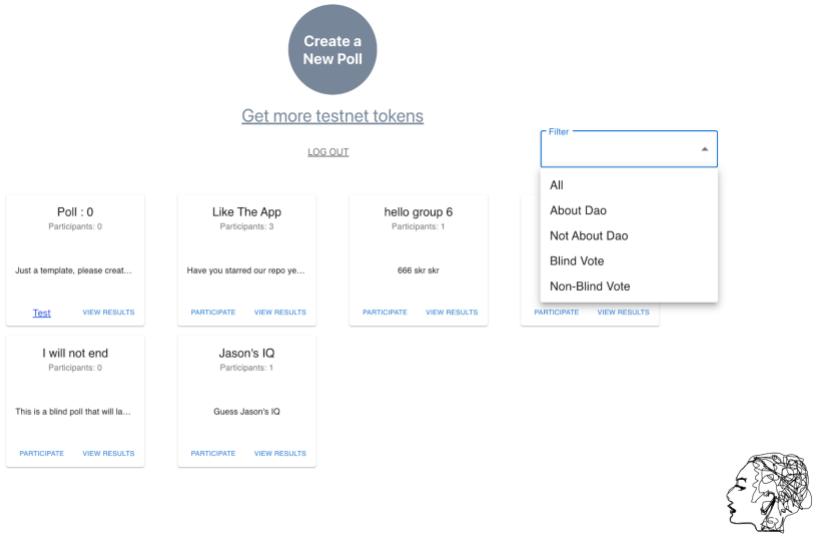

Figure 4. Dashboard Overview

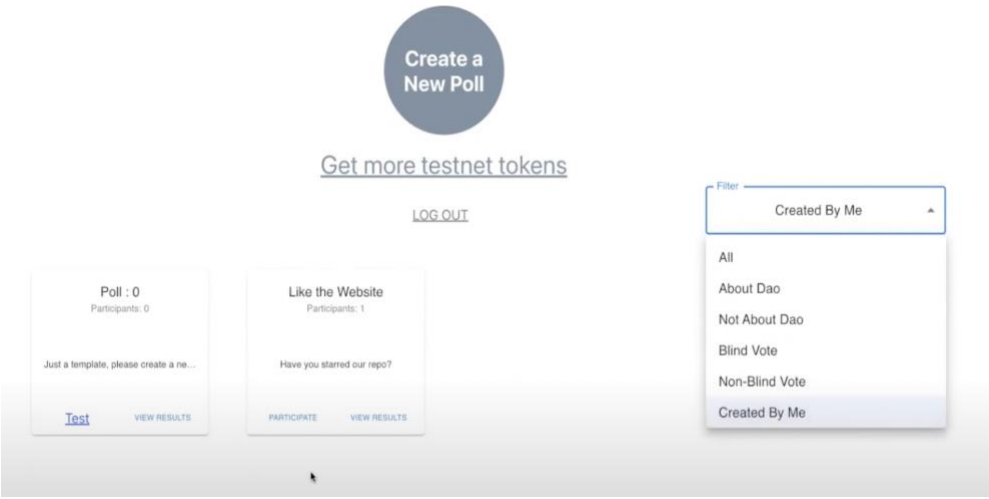

Figure 5. Dashboard View with Filter

6.3 Event Creation Page

This event creation page is designed for users to create and post a new polling event. On this page, users are expected to provide a new poll name, poll duration in minutes, poll description, and selectable options. For each poll, a minimum of two options are required and it can have up to eight options in total. Moreover, every poll could be set to blind vote and/or categorized as a poll about this DAO.

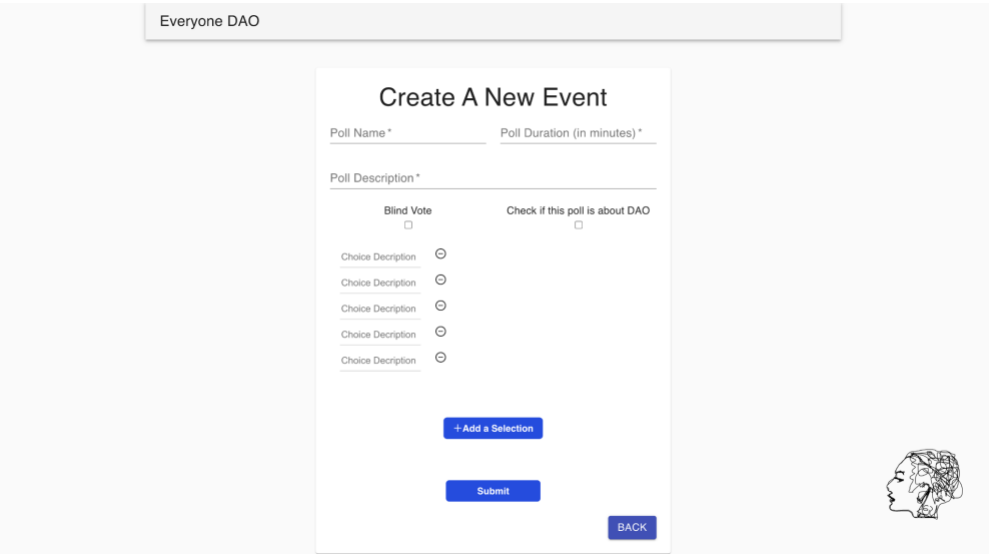

Figure 6. Event creation page

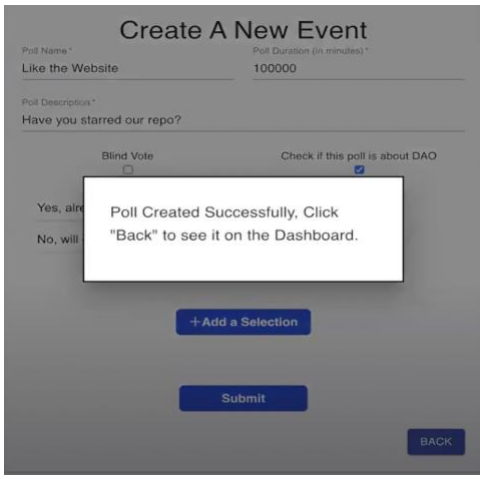

Figure 7. New event created pop up

6.4 Poll Detail Page

A poll detail page consists of a specific poll's information, including a poll name, a complete poll description, a list of option descriptions, and a message to indicate a user's voting history. A user could make a selection on this page and review the results of the poll. After voting his/her choice would be displayed to him/her on the message area. More, this user could revote if he/she changed his/her mind.

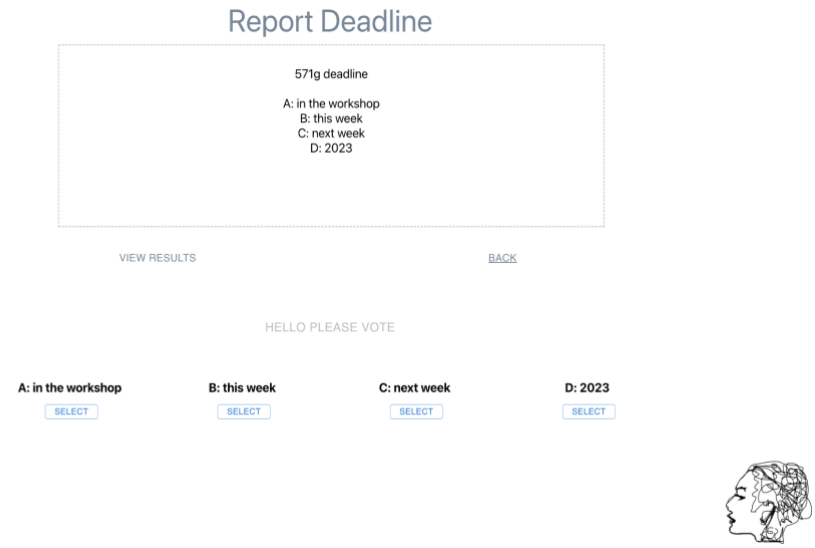

Figure 8. A Overview of a polling event

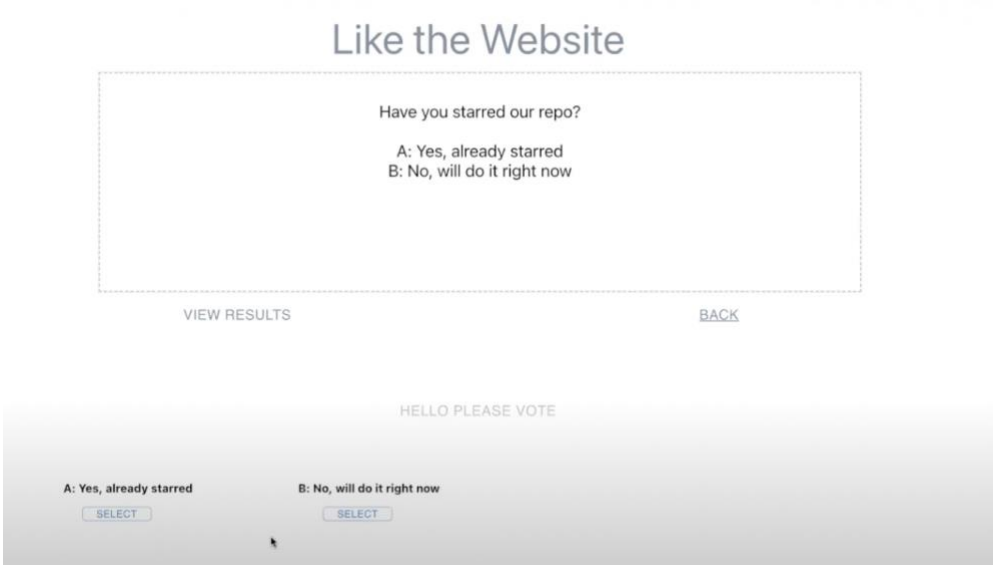

Figure 9. Before voting

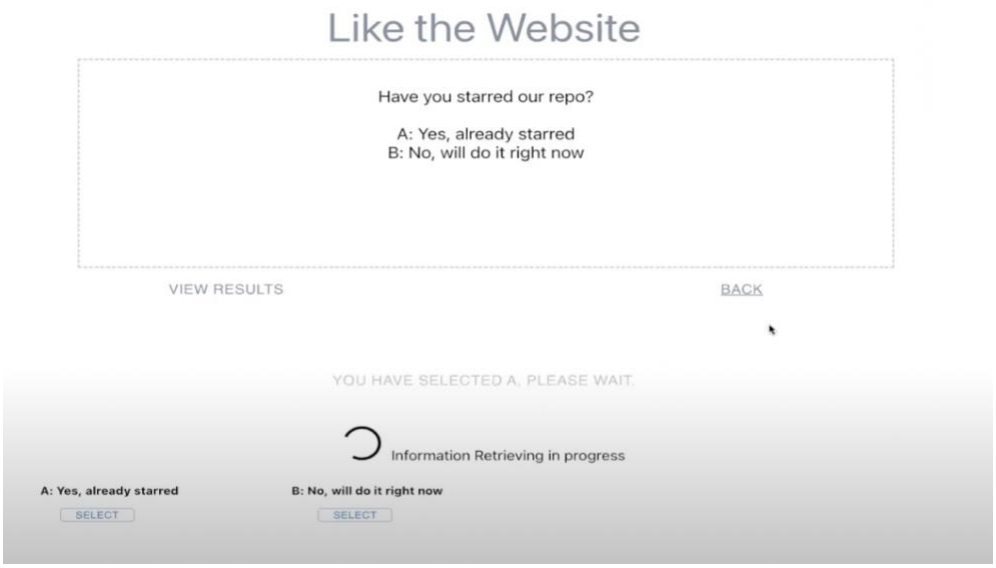

Figure 10. Voting in Progress: Voting for A & waiting for the transaction to sign on the chain

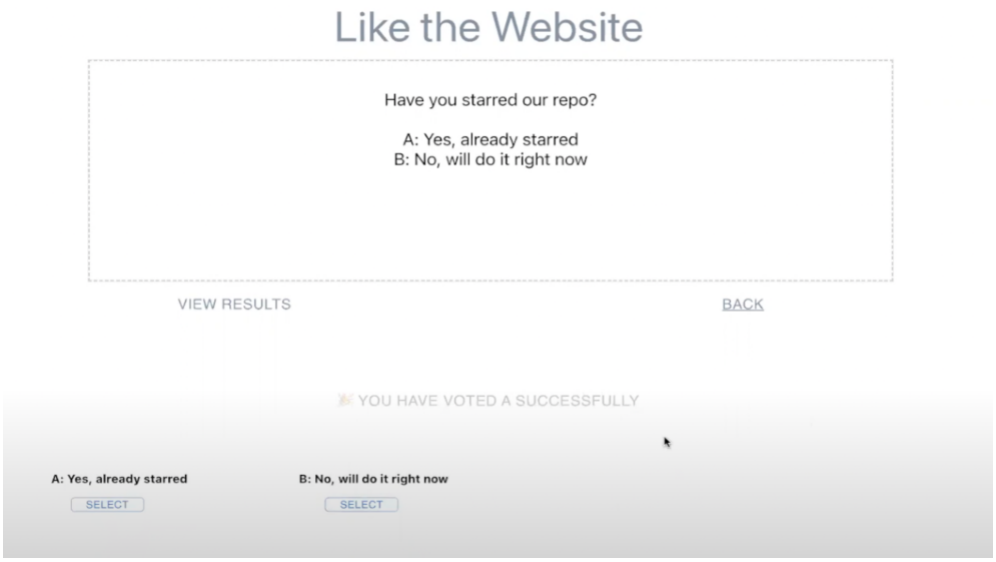

Figure 11. User's selection would be displayed in the center

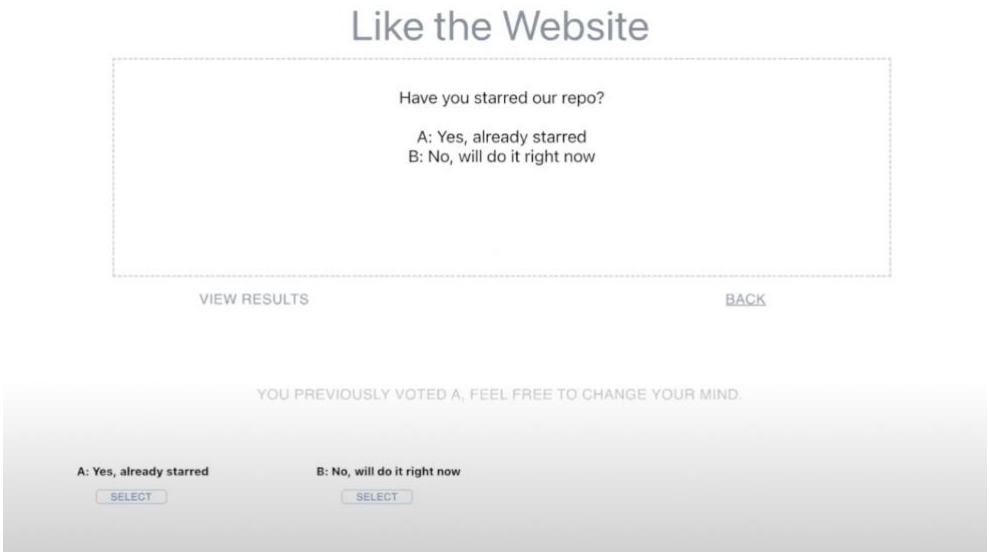

Figure 12. User could also revote before the end of the polling event

6.5 Results Viewing Popups

On results viewed and requested confirmed by the Ethereum blockchain, a pop-up window will be there to show the results. This pop-up window can be accessed from the dashboard page and also the poll detail page. Results would be shown directly if the polling event has ended or if the poll is not blind. Otherwise, if the polling event is blind, users could only view the results when it ends.

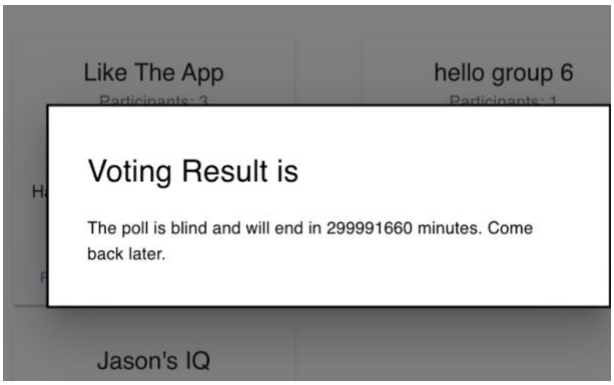

Figure 13. Blind Poll - One could only view the results when the poll has ended

Figure 14. Non Blind Poll - One could view the results in real time even if no one has voted yet

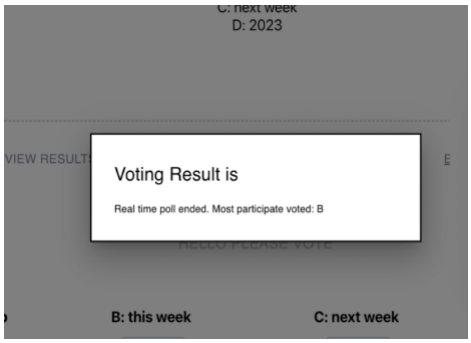

Figure 15. Non Blind Poll - One could view the results after the polling event has ended

7. Discussion and Future Development

7.1 Application Features

Compared to an elementary DAO where there is only one event/ poll that everyone can add proposals to and everyone is able to vote for one proposal, this application allows users to create multiple events with proposals given at the creation of the event. Potential improvement features can be that the choice/description of a poll can be added or edited after a poll is created.

Other than the flexibility of multiple events, this application also supports some interesting features, such as re-vote (Figure 12), autonomously timed stop (see in 5.2), trust-based blind result view (discussed more in 7.3), tag filters in the dashboard (Figure 5), and tie status, which is updated when updated result when voting or reviewing results.

7.2 Code Reflection

There are a few points that the contract code can improve on. First, instead of just reverting a transaction, sending an error in solidity might be a better way to let users understand they are doing an invalid action, such as trying to vote after the vote duration passes.

Second, we tried to pass the information to users with events emission and found that this will trigger information showing in all users' interfaces at the same time because of an event listener. We ended up walking around this by displaying the emitted events only to the user who has an address match with who triggers the emission. This is because, in some private transactions such as vote/ view result, only the user himself/ herself needs to know the transaction happened. Thus, a future improvement could be deleting those event emission functions for private transactions but keeping those for global transactions, such as when a poll is created or ended.

Third, although we have set an “about DAO” tag, we are not really doing anything to upgrade the DAO. Thus, to make a proposed “about DAO” poll really happen, it would be better for the contract to be upgradeable with separate data and logic.

7.3 Blind Vote Implementation

The current blind vote feature is a trust-based one, meaning that although we can not view the results on the website in the front end, the results are actually stored in the blockchain. A hacker who wants to glimpse a blind poll result before it ends can easily peek at it on the chain. A better approach to make the voting blind and binding would be setting one more REVEALING status for the poll, so the statuses would be VOTING, REVEALING, and ENDED. During the voting period, the voting results will not be stored in the chain but only a hashed version of it; then in the REVEALING period, voters will send the unencrypted results again and only the hash values match those stored in the contract, the votes would be verified and stored. In this case, the real results will be logged into the chain only in the REVEALING stage, even if the hacker can see the results in this stage, he/she cannot vote at that time, thus not able to manipulate the vote.

7.4 Cyber Attack vs. Level of Decentralization Trade-off

The voting results might be manipulated by hackers who hold multiple wallets. To prevent the issue, a community-specific NFT can be created and issued to the participants. Only those who have the NFT would be able to register on the platform. However, this will compromise a certain level of decentralization, as assignments of NFTs might come from centralized governance power.

8. References

Democratic Institutions. (2017). Online Voting: A Path Forward for Federal Elections. Retrieved from the Reports and publication of Government of Canada website: https://www.canada.ca/en/democratic-institutions/services/reports/online-voting-path-forward-federal-elections.html#shr-pg0

Jafar, U., Aziz, M. J. A., & Shukur, Z. (2021). Blockchain for electronic voting System—Review and open research challenges. Sensors (Basel, Switzerland), 21(17), 5874. https://doi.org/10.3390/s21175874

Lahane, A. A., Patel, J., Pathan, T., & Potdar, P. (2020). Blockchain technology based e-voting system. ITM Web of Conferences, 32, 3001. https://doi.org/10.1051/itmconf/20203203001

Puneet, Chaudhary, A., Chauhan, N., & Kumar, A. (2021). Decentralized voting platform based on ethereum blockchain. Paper presented at the 1-4. https://doi.org/10.1109/ICAECT49130.2021.9392580

Wang, B., Sun, J., He, Y., Pang, D., & Lu, N. (2018). Large-scale election based on blockchain. Procedia Computer Science, 129, 234-237. https://doi.org/10.1016/j.procs.2018.03.063

Yacoubi, A., Erraha, B., & Asri, H. (2021). An electronic voting system adopting blockchain: Interpretation, characteristics and investigation. E3S Web of Conferences, 297, 1076. https://doi.org/10.1051/e3sconf/202129701076