Electora: Election Management System

13th February 2022 **Team: Coding Pals**

Team Members: Gayathri V, Nachiappan SK, Mrinal M, Manasa K

Abstract

The purpose of an Election Management System is to conduct elections and allow authorized users to vote. Any Election Management System has to make sure the election is conducted in a fair manner and the integrity is maintained. Some of the main challenges of building an Election Management system are ensuring that each user can vote only once, once voted the vote shouldn't be changed (immutability of data), unauthorized users shouldn't be able to use the data and finally the rules of election should remain the same throughout the election. Using a decentralized system allows us to take care of these issues by the very nature of it. Here we have attempted to build a basic Election Management System using Smart Contracts (Etheruem network).

Introduction

We are building this Election Management System as part of the Synergy Hackathon. The structure of this article will be as follows. In the first section we will briefly discuss the requirement of building an Election Management System, in the second section we will discuss how we can leverage blockchain technology to overcome these challenges. The third section will be about the features of the application followed by a detailed description of how we developed the application. The fifth section is about the challenges we faced and the limitations of the system. And finally in the final section we discuss is the Future Work we are planning to do.

1. Requirements of Building an Election Management System

The major requirements of an Election Management System are listed below.

- Ensuring that the Election is not rigged, i.e the election results are not modified by any parties
- Ensuring that only authorized users can be part of the election

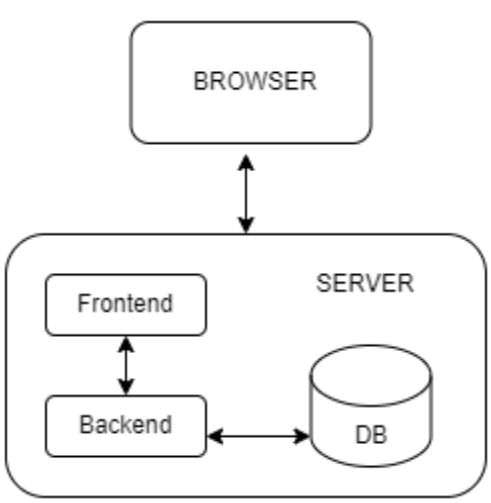
- Ensuring that any authorized user can only vote once
- Ensuring that the rules of the election remains the same throughout the election

Using a centralized system is a challenge to maintain the above requirements, because any third party can hack into the centralized database and alter data like the vote count, rule of the election and hence the integrity of the election becomes compromised. Hence we propose to use a decentralized system to ensure the requirements. How we can use blockchain technology to build a decentralized system is discussed in the next section.

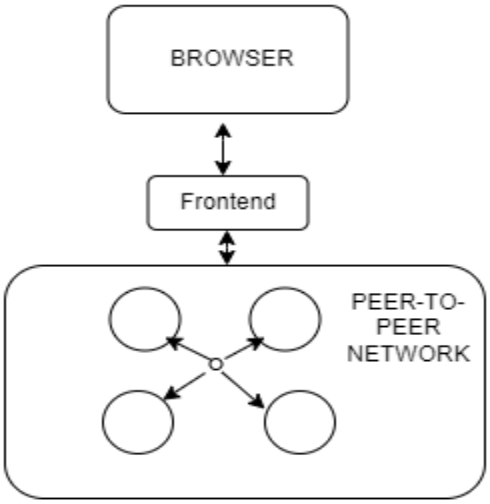
2. Blockchain Technology

Blockchain Technology is a distributed database that is shared among the nodes of the network[1]. There are different applications of blockchain technologies, two of the most popular ones being bitcoin and ethereum. Ethereum allows people to build decentralized applications (known as DAPPS). Here we are building a DAPP for our Election Management System.

Unlike a regular client-server architecture, blockchain follows a peer-to-peer network design. Data does not lie on a single server, but is distributed across different nodes.



Centralized System



Decentralized System

The decentralized nature of the blockchain helps us to ensure that any individual user cannot interfere and make changes to the network. In addition to that, blockchain uses cryptographic hashes which makes the data immutable in nature[2]. These help us to ensure that the results and the rules of the election cannot be tampered.

3. Features of the Election Management System

i) Login/Register Page: The Login or Registration will be using an verifiable id of the user, that depends on the need. For e.g. in a college election the registration id can be the college registration ID and in case of a local body election the id can be the voter id. The unique nature of the registration id along with the wallet id(metamask id) will ensure that an authorized user can only vote once.

ii) Dashboard: The Dashboard page contains two click buttons, one named Elections and the other named Results. Clicking the Election button will redirect the user to the Ongoing Election Page and clicking the Result button will take the user to the Concluded Elections Page.

iii) Ongoing Election Page: This page shows the number of elections that are currently going on. Each of the elections is a clickable button which will take the user to the corresponding Election page.

iv) Concluded Election Page: This page shows the list of all the elections that have already concluded. Clicking on any of the concluded election button will take the user to the corresponding Results Page

v) Election Page: The election page is unique for each of the ongoing elections. It displays two buttons, one is the Candidates Button and the other is the Vote button. The Candidate button will take the user to the Candidate Page whereas the Vote button will take the user to the vote page.

vi) Results Page: The result page is also unique for each of the concluded elections. It displays the Winner of the election and the vote distribution about the candidates.

vii) Candidate Page: The candidate page contains a dropdown of a list of candidates of the selected election. Selecting any candidate will display the details of the candidate.

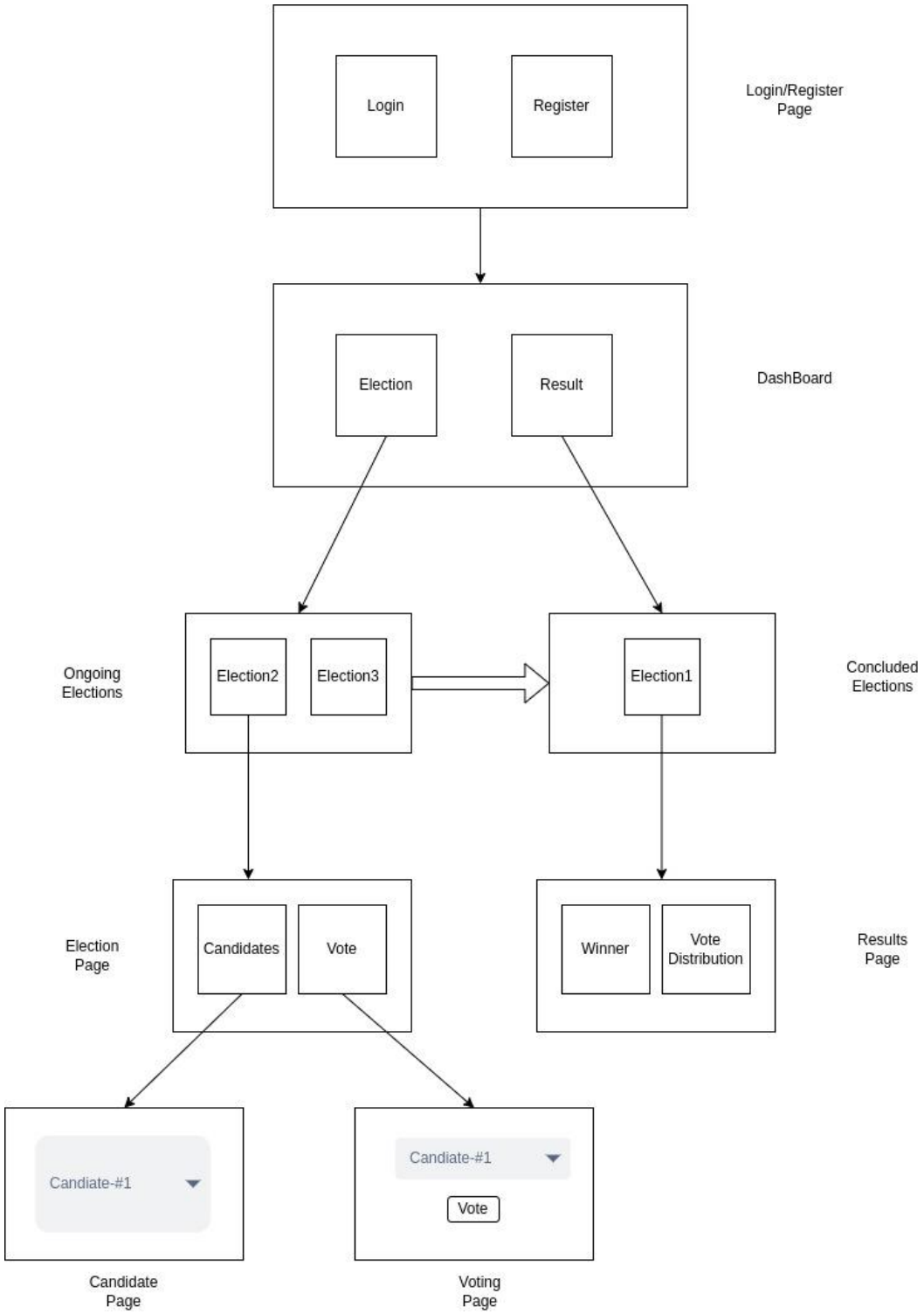
viii) Vote Page: The user can cast the vote from this page. He can select the candidate he wants to vote from the dropdown and click on the vote button. If the user has already voted, it displays the message that the user is not allowed to cast multiple votes.

4. Development

Tech Stack: Solidity for smart contracts, React JS for front-end of our web application, Metamask for ethereum accounts to cast votes.

The back end of our application is the blockchain smart contract written in solidity programming language. We developed the back-end based on the paper “Implementation of Decentralized

Blockchain E-Voting"[3] and youtube video "How to build Ethereum Dapp"[4]. The front end webpage is designed using React JS. The control flow of the front end web-page is given below.



5. Challenges

This is the first time all of us are working on building a decentralized application and hence it was a challenge to understand how blockchains work and how we can use smart contracts to deploy applications. Once we understood how the backend works, then the major challenge was to integrate the backend with the front-end. We had to find suitable dependencies which are compatible with both solidity as well as react JS.

We overcame the challenges by splitting the work efficiently according to the strengths of the team members. We did extensive research to understand the underlying concepts of blockchains and decentralized networks. Along with that, thinking through the design of our application beforehand helped us to go about it more efficiently.

6. Future Work

The final application we developed is far from complete, due to the time constraint, we only implemented the system to handle a single election at a time, extending it to handle multiple elections is one of the major work that remains to be done. Further the application could use elaborate data visualization tools to display various statistics of already concluded elections. Another feature that could be added is a push-notification feature that notifies users at the end of the election about the winning candidate and the vote distribution.

References:

[1] https://www.investopedia.com/terms/b/blockchain.asp

[2] https://www.upgrad.com/blog/what-makes-a-blockchain-network-immutable/

[3] https://eudl.eu/pdf/10.4108/eai.13-7-2018.164859

[4] https://www.youtube.com/watch?v=3681ZYbDSSk