Electoral Voting on Blockchain

CS-2361
Course Project
https://github.com/manyasachdev/cs2361_final_project

Manya Sachdev

Motivation Behind The Project

Traditionally, voting took place by means of paper ballots. Up until the 1990s, every eligible individual would have to manually mark their desired candidate on a piece of paper and then place this paper in a locked box. The simple nature of this method made it prone to attacks and fraudulent practices such as stuffing fake ballots into a box. This led to casting of fake votes with no traceable means to connect vote to voter.

The introduction of EVMs did not solve this problem. There is currently an upwards trend seen in the number of cases of corrupt EVMs. Despite being initially regarded as malpractice proof, we have seen how certain larger and more influential political parties tamper with the technology and use EVMs to their advantage.

This project attempts to use blockchain’s features to overcome some of the shortcomings in the current voting system in an effort to promote fairer voting practices.

Advantages of using a decentralised Voting Platform

- Anonymity in voting

  In a decentralised voting application based on the blockchain, voters will have the ability to choose a candidate without any eyes on them. Votes will be mapped to a transaction id and no other identifying key. This will protect those intimidated to vote a particular way.

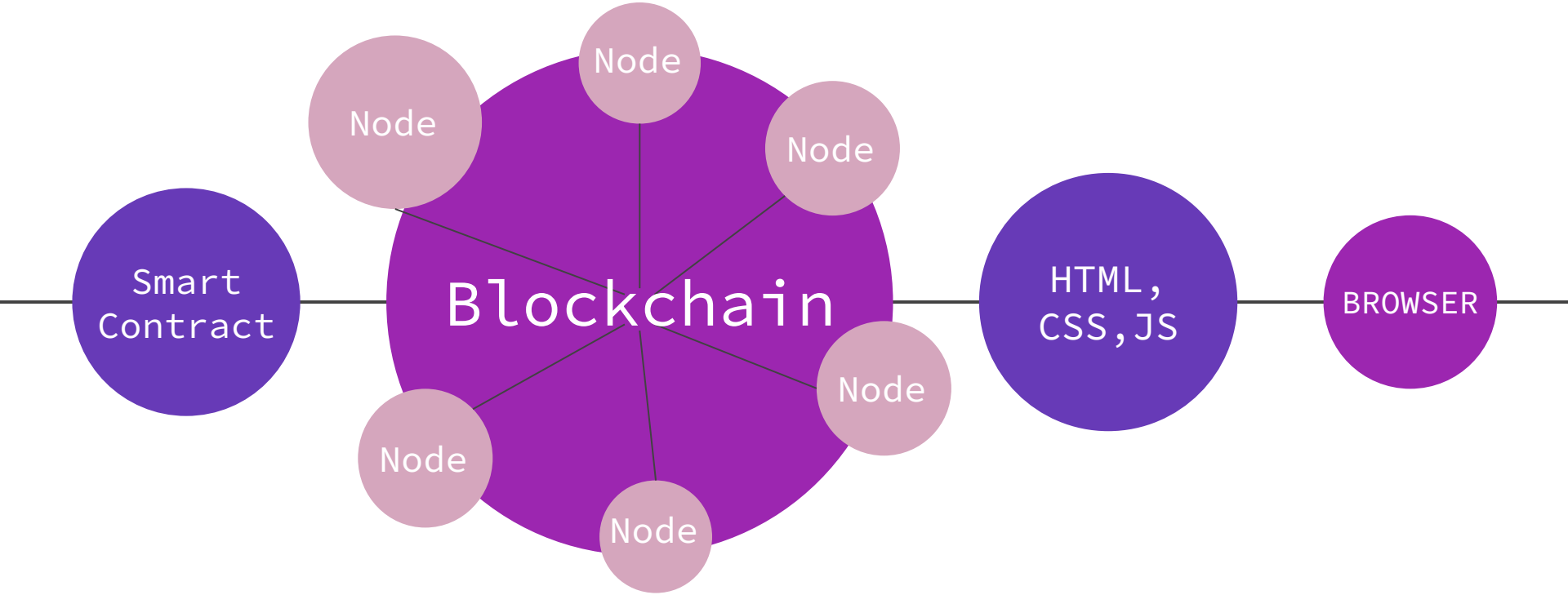
- Transparent and public voting history

  The blockchain is updated in plain site after every transaction. Any individual with access to the chain will be able to view every transaction, i.e. every vote cast.

- Immutable transactions

  Consensus protocol makes it hard if not impossible for hackers to gain access into the blockchain and change votes that have already been cast and recorded.

Basic Structure



The Smart Contract

Election.sol

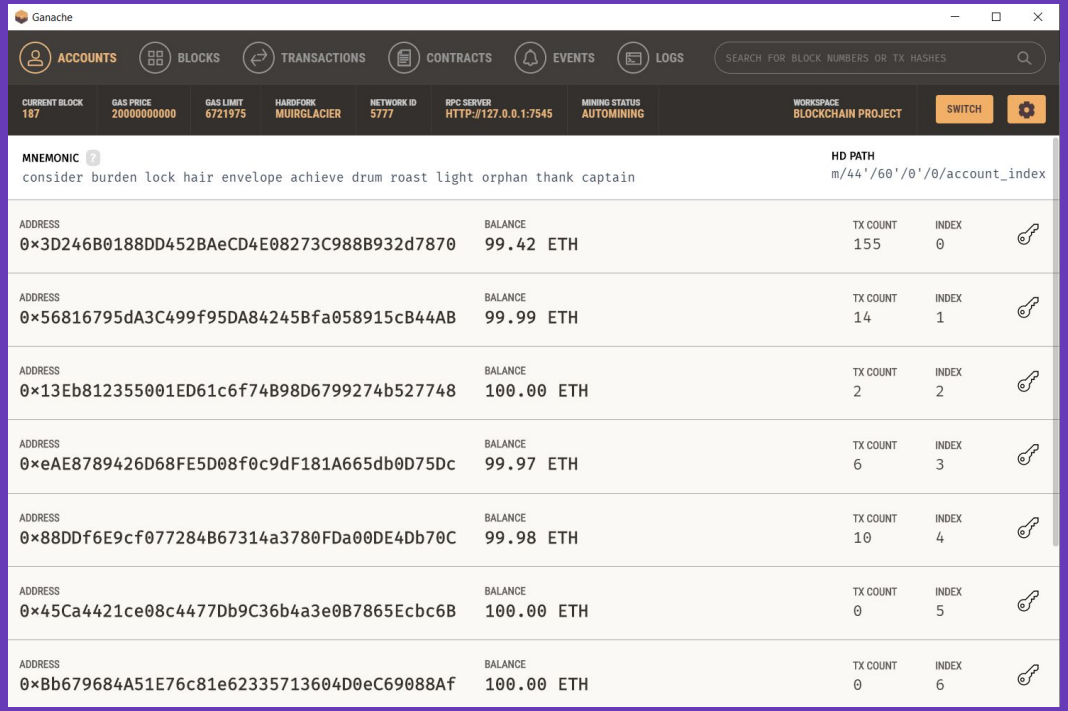
- The contract defines a structure **Candidate** which stores a candidate's unique id number, their name and the number of votes casted in their favor.
- It contains 4 functions in total:
  - **function addCandidate()** which takes a string parameter as an argument. This is the candidate's name. The number of votes are set to 0 by default.
  - **function vote()** which accepts a unique id from the voter to identify which candidate's vote count to increment. Additionally, it adds the voter's address to a mapping which helps us prevent double voting.
  - **function start_election()** and **function end_election()** which can only be called by the owner of the contract. They set the value of an arbitrary variable **status_election** to either 1 or 2 to indicate whether voting is open or closed respectively. The value of this variable is 0 by default.

```
pragma solidity >=0.4.22 <0.8.0;
//pragma solidity 0.7.6;

contract Election {
    uint private status_election; //0-not started, 1-ongoing, 2-finished
    address public owner;

    struct Candidate {
        uint id;
        uint num_votes;
        string name;
    }

    uint public num_candidates;
    mapping(address => bool) public voters;
    mapping(uint => Candidate) public candidates;
    mapping(address => uint) public voted_for;


    constructor () public {
        owner = msg.sender;
        status_election = 0;
        addCandidate("ABC");
        addCandidate("XYZ");
    }

    function addCandidate (string memory _name) private {
        num_candidates++;
        candidates[num_candidates] = Candidate(num_candidates, 0, _name);
    }
```

```

    function vote (uint _candidateId) public {
        require(status_election == 1);
        require(!voters[msg.sender]);
        require(_candidateId > 0 && _candidateId <= num_candidates);
        voters[msg.sender] = true;
        voted_for[msg.sender] = _candidateId;
        candidates[_candidateId].num_votes ++;
    }

    function start_election () public {
        require(msg.sender == owner);
        status_election = 1;
    }

    function end_election () public {
        require(msg.sender == owner);
        status_election = 2;
    }
}
```

Local Blockchain

Ganache



Front End

Voting!

SNo.	Name	Votes
1	ABC	0
2	XYZ	0

Select Candidate

ABC

Vote

Your Account: 0x88ddf6e9cf077284b67314a3780fda00de4db70c