
Lab 1

CSE 526: Blockchain

Ved Harish Valsangkar
Person Number: 50290388
vedharis@buffalo.edu

Smrati Kushwah
Person Number: 50292824
smratiku@buffalo.edu

Overview

The state of Maine has approved the use of ranked voting in the state elections. Each voter ranks the nominees for a post based on his preference. the winner is a nominee with a clear majority of the votes with him/her as the first rank. If this is not achieved, the nominee with the lowest first rank is eliminated and voters who liked that candidate the best have their ballots recounted for their second choice. This process repeats and last-place candidates lose until one candidate reaches a majority and wins. Voters have the right to abstain from voting as well for political reasons.

Design Diagrams

Contract MSRVE_Ballot
<pre>struct Voter uint256 numProposals address [] voterList enum Phase {Regs, Vote, Done} Phase public state = Phase.Regis address admin uint256[] runningCount uint public voteCount uint public abstainCount uint256 public winner bool calc mapping(address => Voter) voters</pre>
<pre>modifier canVote() modifier onlyAdmin() modifier validPhase(Phase status)</pre>
<pre>constructor(int numProposals, int numCandidates) public register() public validPhase(Phase.Init) { } abstain() public canVote(msg.sender) validPhase(Phase.Open) { } vote(uint[] preference) public canVote(msg.sender) validPhase(Phase.Open) { } delegate(address to) public canVote(msg.sender) validPhase(Phase.Open) { } calcWinner() public validPhase(Phase.Close) onlyAdmin { } getWinner() public view { }</pre>

Figure 1: Contract Diagram

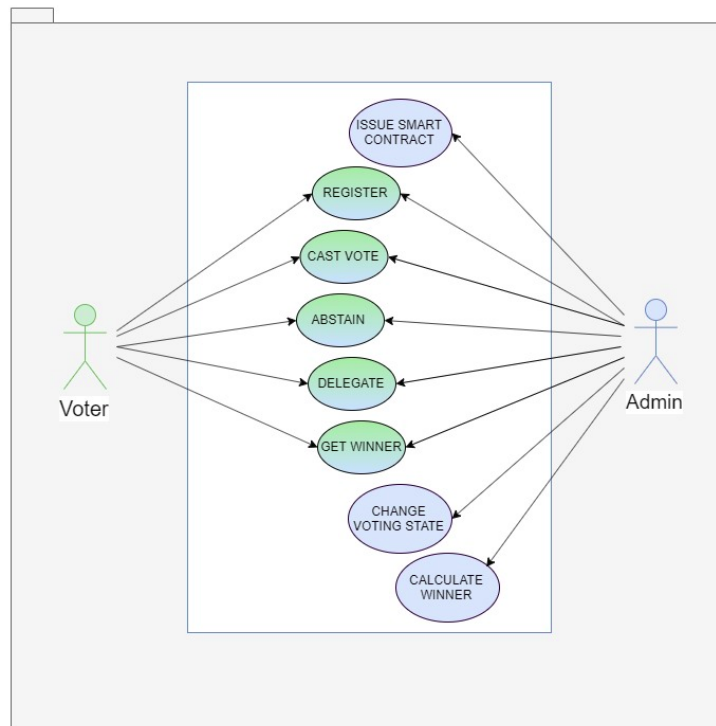


Figure 2: Use Case Diagram

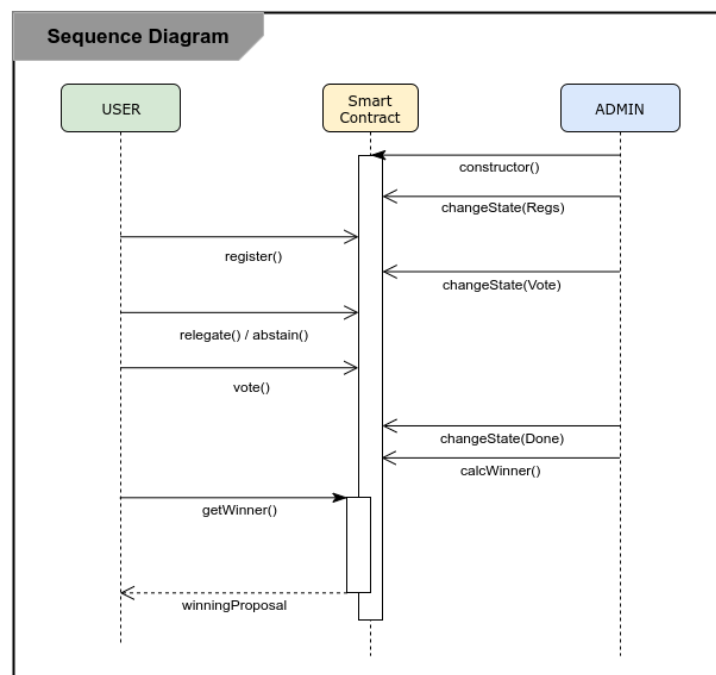


Figure 3: Sequence Diagram

Workflow

Structure

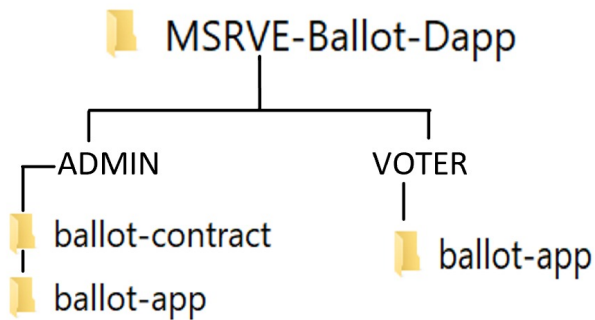


Figure 4: Access Allowed per Role.

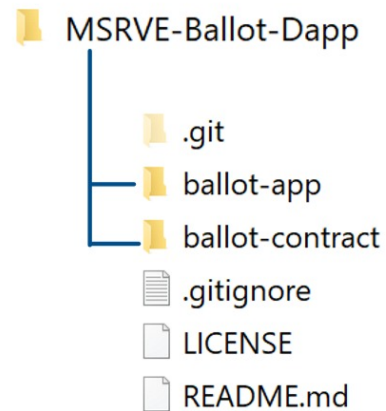


Figure 5: Project Top Level Directory Structure

As seen in Figure 4, the admin has access to the app and the contract folders. As per the current system the admin has to manually change the system state and trigger calculation of the winner using `truffle console`.

Walk-through with Screenshots

These are the steps to follow:

Step 1 Enter username and password to login to the Dapp. User names and passwords can be found in `MSRVE-Ballot-Dapp/ballot-app/db.json`.

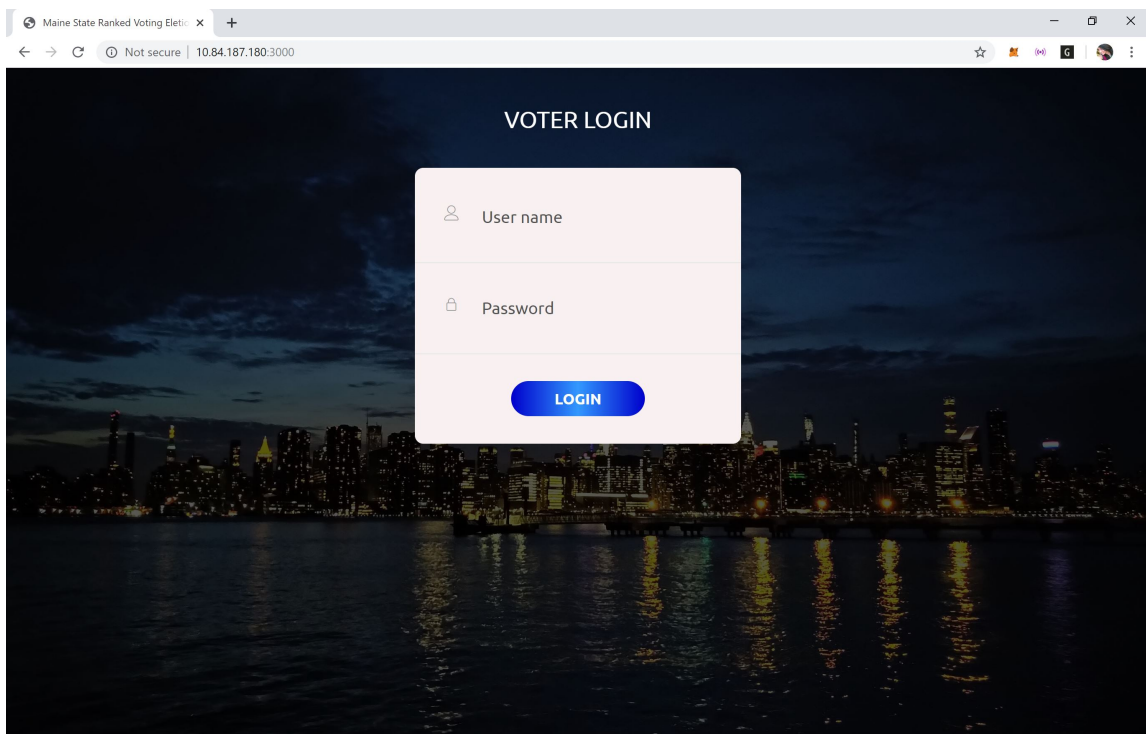


Figure 6: Login page on accessing localhost:3000

Step 2 After login, you will see the Dapp mainscreen.

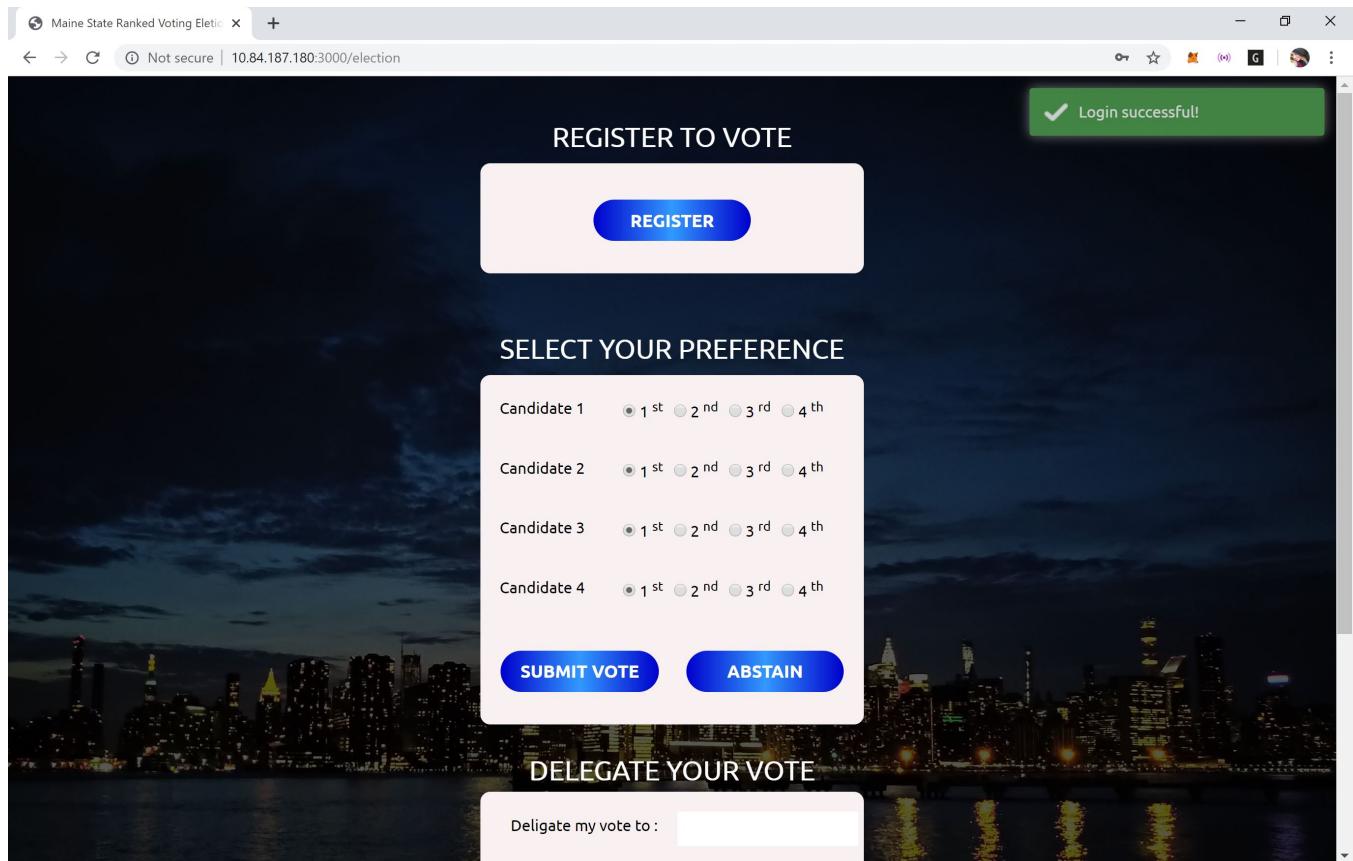


Figure 7: Login Successful

Step 3 Click on register. Users need to register for their vote to be counted.

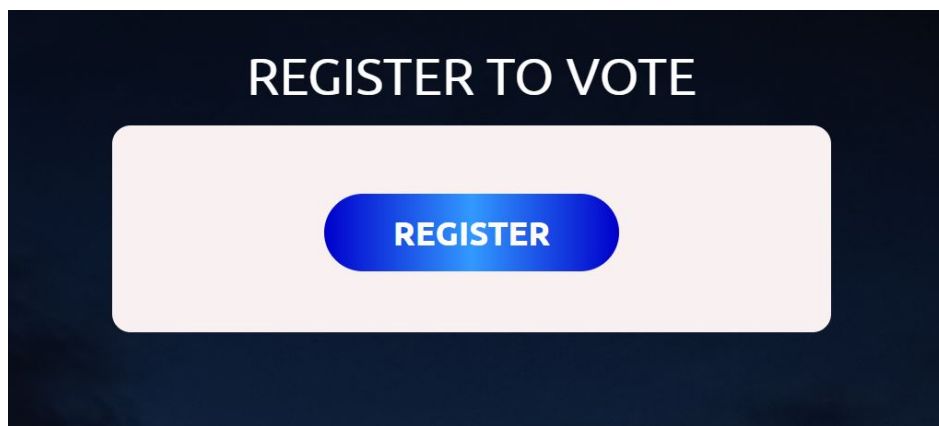
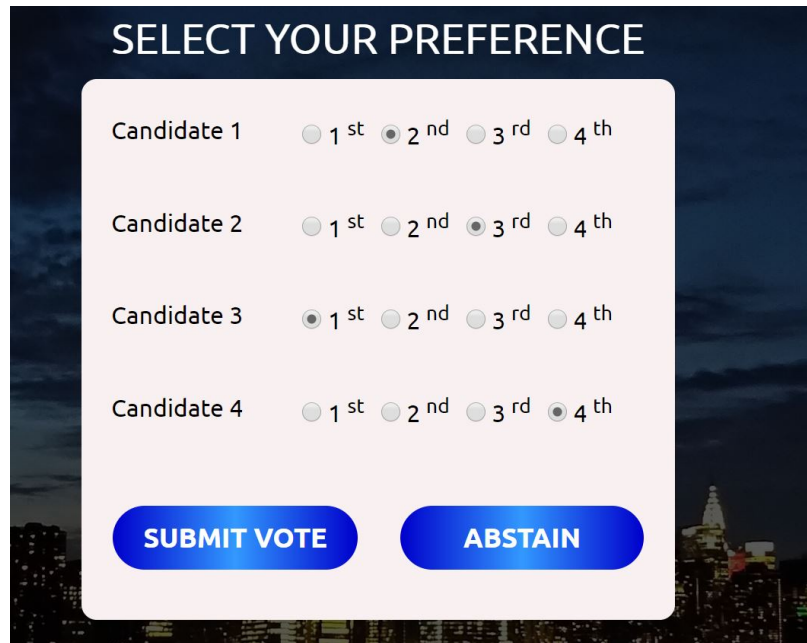


Figure 8: Click on Register to register user

Step 4 When the Admin changes state to VOTE, users may cast their vote. They may choose to abstain from voting in this election or delegate their vote (next step).



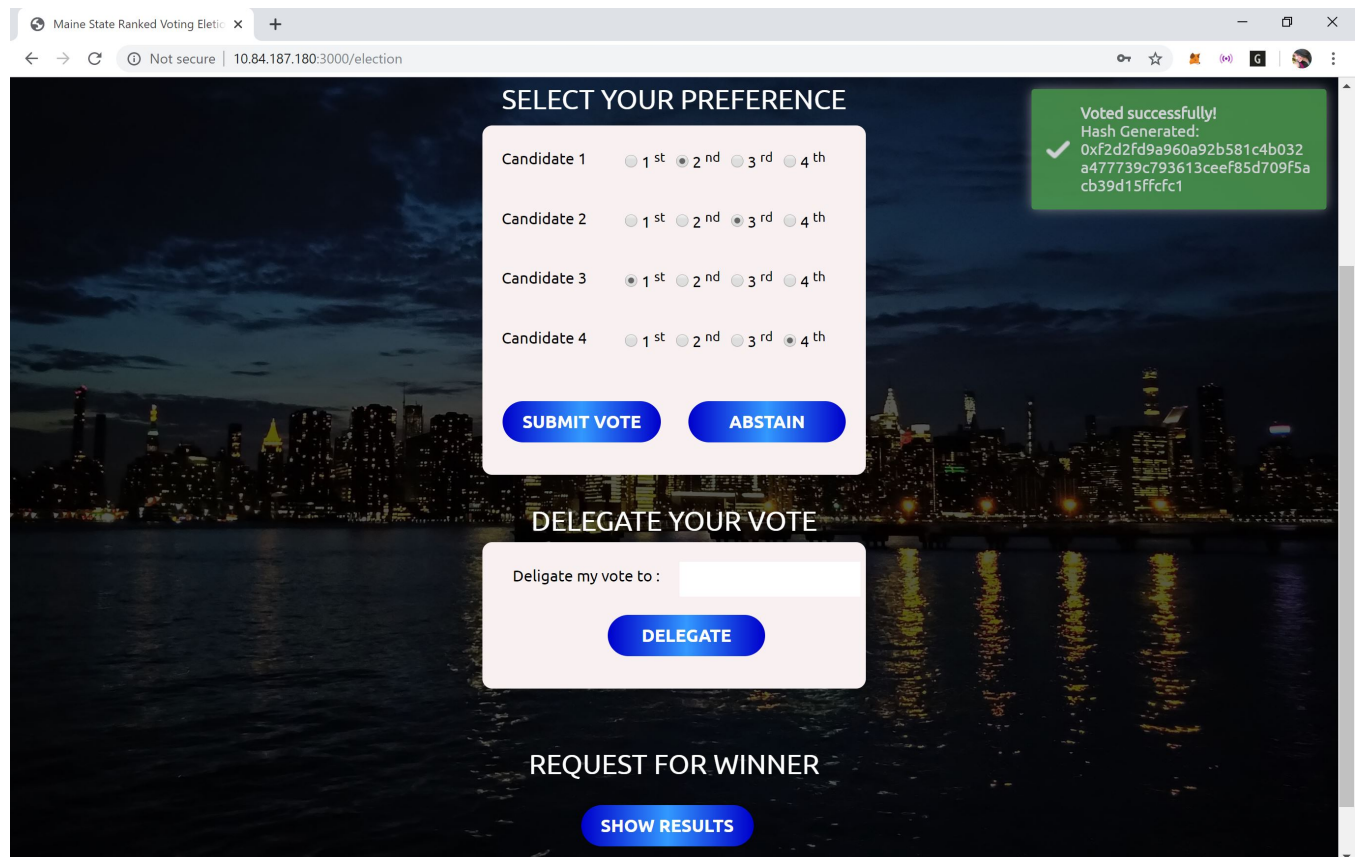
The screenshot shows a web interface titled "SELECT YOUR PREFERENCE" with a dark city skyline background. It contains four rows of radio button options for ranking candidates:

- Candidate 1: ☐ 1st ☒ 2nd ☐ 3rd ☐ 4th
- Candidate 2: ☐ 1st ☐ 2nd ☒ 3rd ☐ 4th
- Candidate 3: ☒ 1st ☐ 2nd ☐ 3rd ☐ 4th
- Candidate 4: ☐ 1st ☐ 2nd ☐ 3rd ☒ 4th

At the bottom are two blue buttons: "SUBMIT VOTE" and "ABSTAIN".

Figure 9: Select desired ranks and click on Vote. Alternatively, click on Abstain to abstain

Step 4.1 A Toast is displayed on successful execution of the Vote, Abstain or Delegate transaction. If the transaction was denied or error was thrown in the blockchain, that error is also displayed in Toast of red background.



This screenshot shows the same "SELECT YOUR PREFERENCE" interface as Figure 9, but with additional sections and a toast notification. Below the candidate rankings are two buttons: "SUBMIT VOTE" and "ABSTAIN". Below these are two more sections:

- DELEGATE YOUR VOTE**: A form with the label "Deligate my vote to:" followed by a text input field and a blue "DELEGATE" button.
- REQUEST FOR WINNER**: A blue button labeled "SHOW RESULTS".

A green toast notification is visible in the top right corner, indicating a successful vote:

```
Voted successfully!  
Hash Generated:  
0xf2d2fd9a960a92b581c4b032  
a477739c793613ceef85d709f5a  
cb39d15ffcfc1
```

Figure 10: The hash of the transaction from the blockchain is displayed as a Toast on successful vote. It is similar for Abstain and Delegate as well.

Step 4.2 To delegate your vote, following conditions must be met:

- User must be eligible to vote and have not cast a vote or abstained yet.
- User must not enter his/her own name.
- The user to whom the vote is delegated must not have voted.
- The user may delegate the vote to a user who has already delegated his/her vote following a chain pattern as long as the end recipient of the delegation has not yet voted or is not the user him/herself.

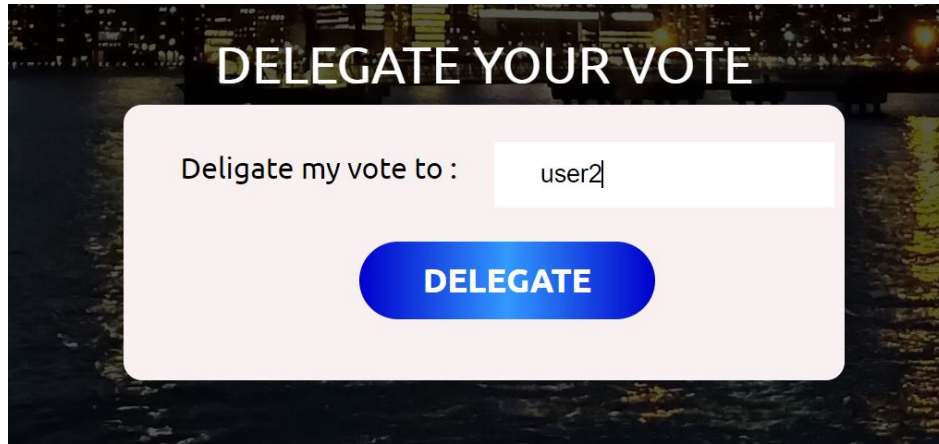


Figure 11: Vote may be delegated to the username entered in the textbox

Step 5 The winner needs to be calculated first at the end of the election. The state change is done and the calculation is triggered by the admin. After this state change, any user (including admin) can request the winning candidate as many times as desired. The winning candidate is displayed below the button and in a Toast.

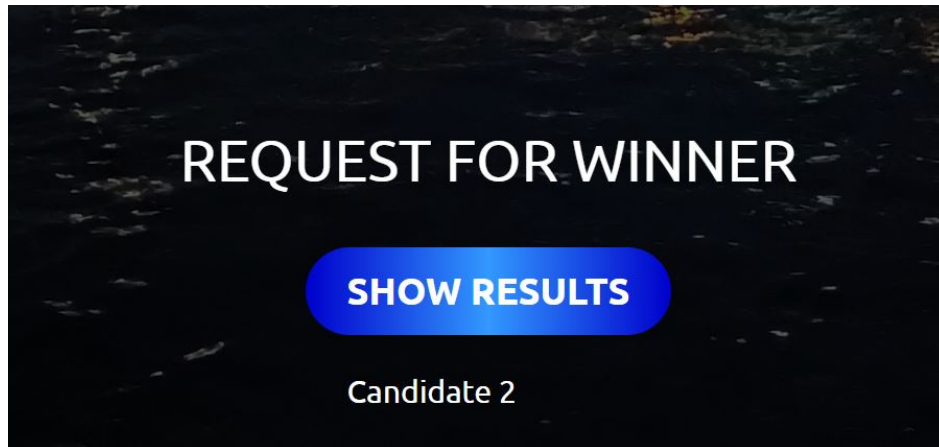


Figure 12: When the winner is calculated at the end of the

Instructions

Following steps are to be followed to install all node modules, test the Smart Contract and run the project. It is assumed that you have Ganache GUI installed. On successful deployment, open page <http://localhost:3000> to access the Dapp. This project can also be downloaded by cloning from GitHub using the link below:

<https://github.com/vedvalsangkar/MSRVE-Ballot-Dapp.git>

Setup

1. Unzip archive `MSRVE-Ballot-Dapp.zip` and enter folder `MSRVE-Ballot-Dapp`.
2. Run command `npm run setup` to install all required node modules at their appropriate locations.
 - (a) Alternatively, navigate to folders `ballot-app` and `ballot-contract` and run command `npm install` at both locations.
3. Open Ganache GUI. Initiate a development blockchain.
4. Navigate to folder `MSRVE-Ballot-Dapp/ballot-contract/migrations` and modify JavaScript files by replacing the address there with the address of admin account (the address of the **last user** from Ganache GUI).
5. Run command `npm start` in folder `MSRVE-Ballot-Dapp/ballot-contract` which shall compile and migrate the Smart Contract.
6. Usernames and password for testing the Dapp can be found in the JSON file `MSRVE-Ballot-Dapp/ballot-app/db.json`.

Run

1. Run command `npm run demo` in folder `MSRVE-Ballot-Dapp` or `MSRVE-Ballot-Dapp/ballot-app` to start the demo run.

OR

1. Navigate to `MSRVE-Ballot-Dapp/ballot-contract` and run command `npm start`.
2. Navigate to `MSRVE-Ballot-Dapp/ballot-app` and run command `npm start`.

Test

1. To use the testing framework and test the Smart Contract, run command `npm test` in folder `MSRVE-Ballot-Dapp` or `MSRVE-Ballot-Dapp/ballot-contract` to start the test run.

Admin privilege functions

To change states and to trigger the final vote calculation, run the following steps:

1. Navigate to `MSRVE-Ballot-Dapp/ballot-contract` and run command `truffle console`. Run all following commands in this console.
Note: It is assumed that the admin is set as the last user from the Ganache user list as stated in Step 4 of Setup.
2. Enter command `var ballot = await MSRVE_Ballot.deployed()` to get the deployed contract.
3. Enter command `var acc = await web3.eth.getAccounts()` to get the accounts available in Ganache.
4. To change the Smart Contract state, run command `ballot.changeState(<state>, from:acc[9])`, where `<state>` is the new state. 0 is for Registration, 1 for Voting and 2 for Done state.
5. To calculate the final winner, run command `ballot.calcWinner(from:acc[9])`.