**Voting with Blockchain**

**Problem Statement:**

The election is a fundamental pillar of a democratic system, which enables the public to express their views in the form of a vote. Due to their significance in our society, the election process should be transparent and reliable to ensure participants of its credibility.

Decentralization reduces the risk of single points of failure and makes it harder for malicious actors to manipulate election results. Each node in the decentralized network contributes to the security and integrity of the voting process. Online voting is a trend that is gaining momentum in modern society. It has great potential to decrease organizational costs and increase voter turnout. It eliminates the need to print ballot papers or open polling stations—voters can vote from wherever there is an Internet connection. Despite these benefits, online voting solutions are viewed with a great deal of caution because of:

* Eligibility: Only legitimate voters should be able to take part in voting.
* Nonreusability: Each voter can vote only once.
* Privacy: No one except the voter can obtain information about the voter’s choice.
* Fairness: No one can obtain intermediate voting results.
* Soundness: Invalid ballots should be detected and not taken into account during tallying.
* Completeness: All valid ballots should be tallied correctly.

Blockchain technology came into the ground to overcome these issues. It offers decentralized nodes for electronic voting and is used to produce electronic voting systems for their end-to-end verification advantages. This technology is a replacement for traditional electronic voting solutions with distributed, non-repudiation, and security protection characteristics.

In this use case, three parties are involved:

1. Voting Admin

* Start election
* End election
* Show results

1. Voter

* Register a new voter
* Vote for the candidate
* View voter details

1. Candidate

* Register a new candidate
* View the list of candidates

**Features of the application:**

1. Add a new candidate

This function helps to add a new candidate to the election, which can be done only by the admin before the election starts. This function takes the following parameters:

* string memory \_name 🡪 Name of the candidate
* ID 🡪 ID of the candidate(Here ID is given automaticly by the sequence of enroll)

1. Add a new voter

This function helps to add a voter, which can be done only once by the admin before the election starts. This function takes the given parameters:

* address \_voter 🡪 Ethereum address of the voter
* address owner 🡪 Owner of the contract

1. Start Election

This function helps the admin to start the election (setting the Election state to ONGOING). This function takes the address of the contract owner as an input parameter.

1. Display the candidate details

This function helps to show candidate details. The input to this function is the ID of the candidate, and in the response, it returns the following parameters:

* ID 🡪 ID of the candidate
* Name 🡪 Name of the candidate

1. Show the Winner of the election

This function helps to show the winner of the election. This function has no input arguments, but it returns the following fields:

* Candidate name
* Candidate ID
* Votes secured

1. Cast the vote

This function helps voter to cast their vote. It has the below two input arguments:

* Candidate ID is the candidate who has to be voted
* Voter address

1. End the election

This function helps the admin to end the ongoing election. This function can be called only by the admin (contract owner).

1. Show election results (candidate wise)

This function helps to show the votes received by any given candidate. This function takes the candidate ID as input and returns the below fields in response.

* Candidate ID
* Candidate name
* Number of votes received

1. View the voter’s profile

This function helps to view the voter profile. It takes the voter’s address as input and returns the following fields in response:

* Voter’s name
* The candidate who has been voted
* If the vote is delegated or not

**Recommended technologies:**

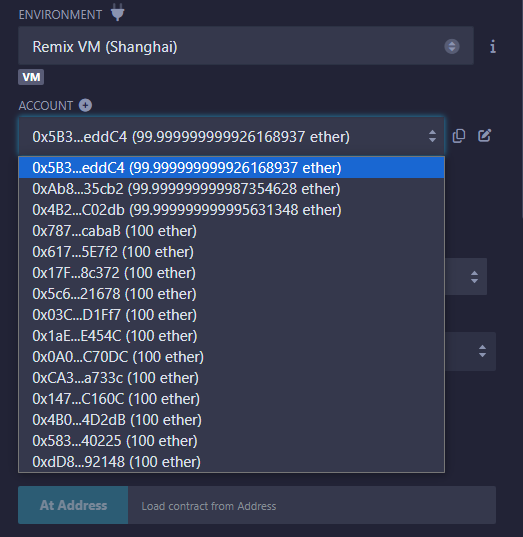
1. Smart Contract development: Solidity
2. IDE Tool: Remix
3. Blockchain: Ethereum
4. Server: Ganache Blockchain

**Project development guidelines:**

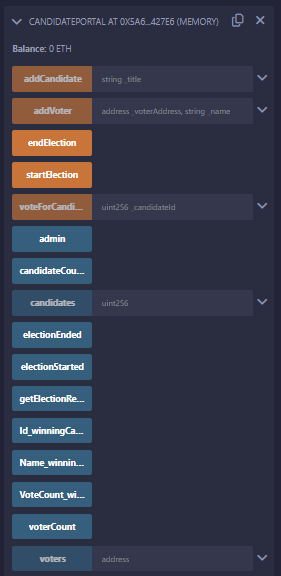
* The project will be delivered within four sprints with every sprint delivering a minimal viable product.
* It is mandatory to do proper sprint planning with user stories to develop all the components of the project.
* The learner can use any technology from the above-mentioned technologies for different layers of the project.
* The learner has to maintain the version of the application over GitHub and every new change should be sent to the repository.
* The learner should also deploy and host the application on any blockchain instance.

**Project output:**

Setting of Wallet Address: Before & after Deployment, the wallet address has to be maintained.



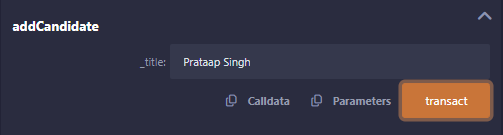
Once you deploy the contract using remix IDE, you can see all the below functions:



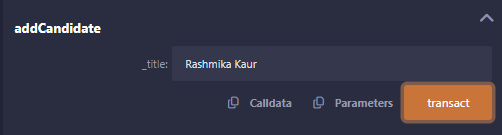
Start the election by clicking :

C:\Users\LENOVO\Desktop\voter\screenshot\Screenshot 8.png

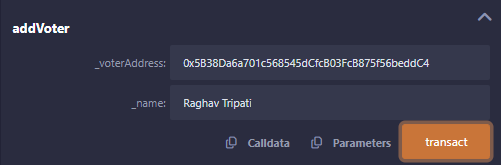
Add a First candidate and transact:



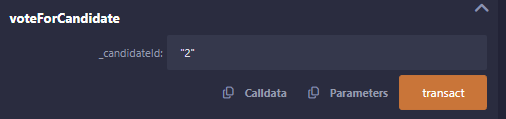
Add a Second candidate and transact:



Add a new voter by giving the wallet address and transact:



Vote for candidate:

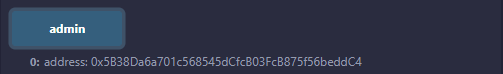


Here vote is given to the Second candidate id “**2**” .

End the election by clicking :

C:\Users\LENOVO\Desktop\voter\screenshot\Screenshot 13.png

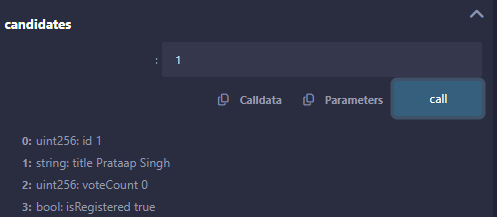
Admin call:



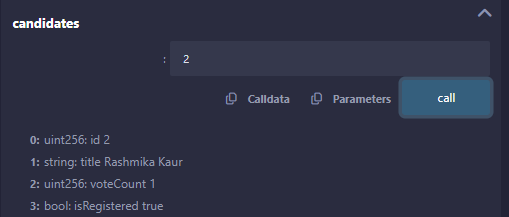
Count the number of candidates participating in the election:



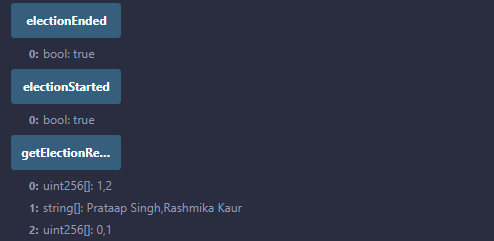
View details of First candidate by entering the id **“1”** and calling:



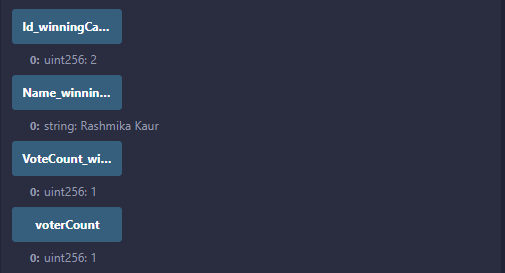
View details of Second candidate by entering the id **“2”** and calling:



Check of Election status and Results : Click Sequence wise



Get **Winning Candidate**: Click Sequence wise



Here the winning candidate is **“ 2 Rashmika Kaur ”.**

View the voter’s profile by entering the wallet address:

