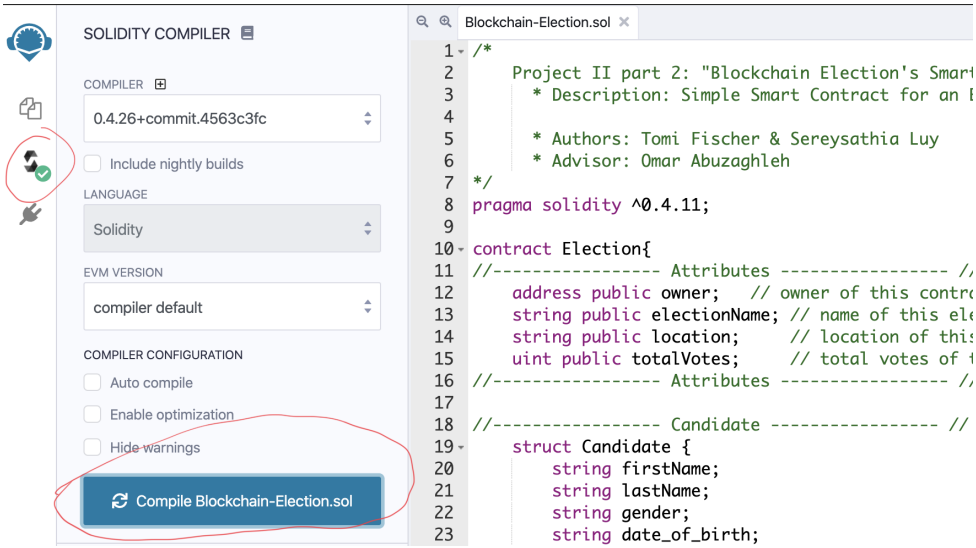
Project: "E-voting using Blockchain Technology - Smart Contract"
Authors: Sereysathia Luy
Advisor: Omar Abuzaghleh
Description: Smart contract for a simple e-voting application using Blockchain Technology.
Programs/IDE used: Ganache, myEtherWallet, and Solidity

How to Run

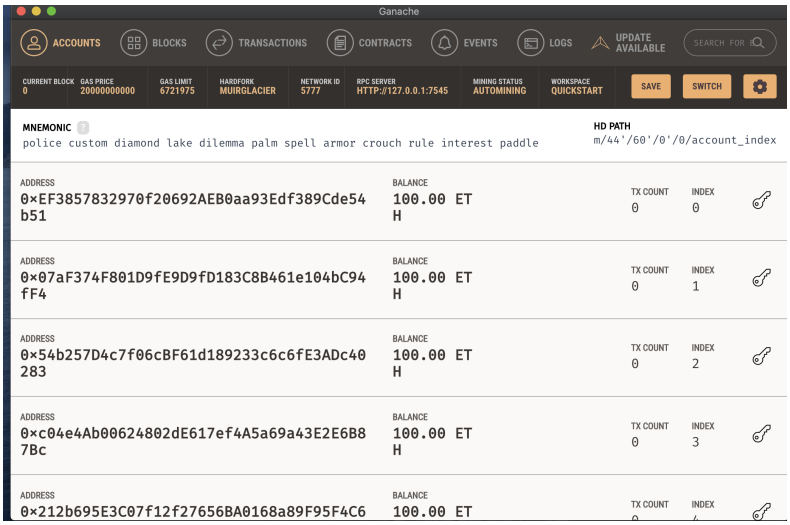
1. Compile the code

- Open Solidity IDE, compile the code



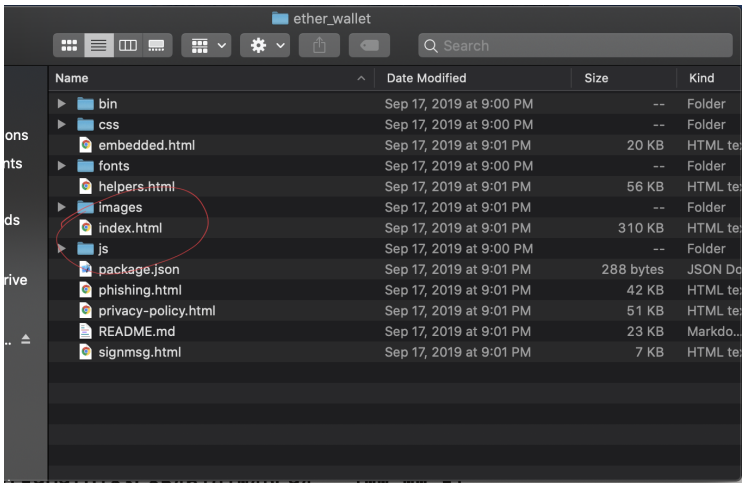
2. Open Ganache

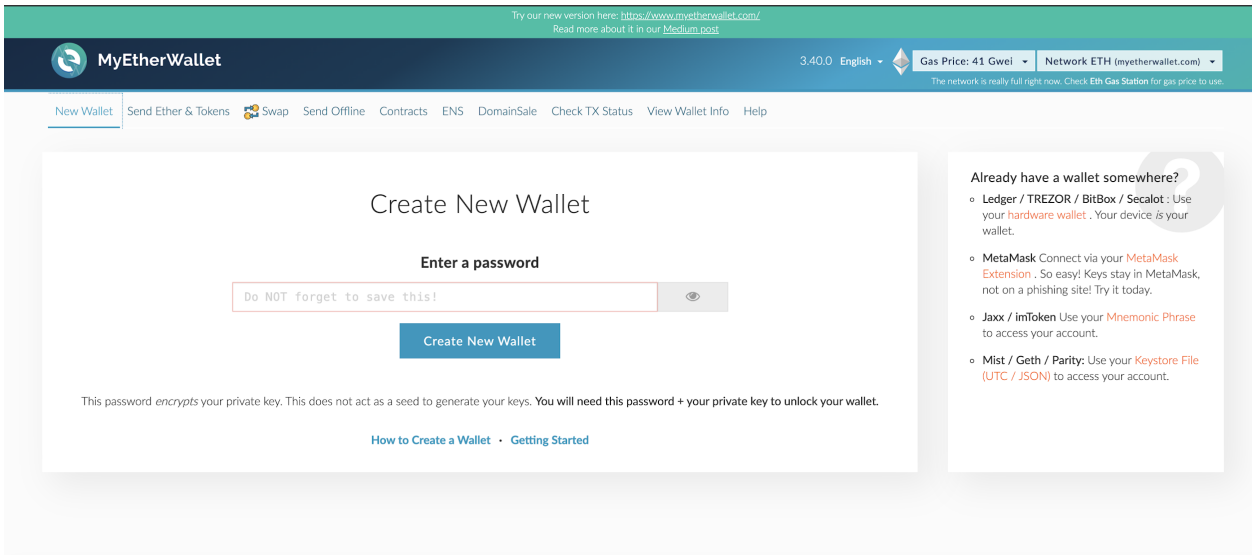
- Open Ganache app -> Quick Start, and you should see this:



3. Open myEtherWallet

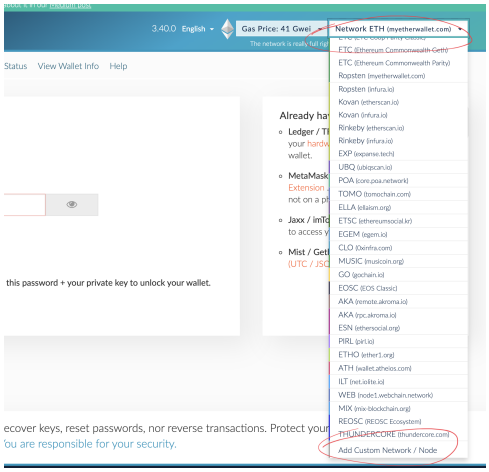
- Open myEtherWallet
  i. Using index.html file



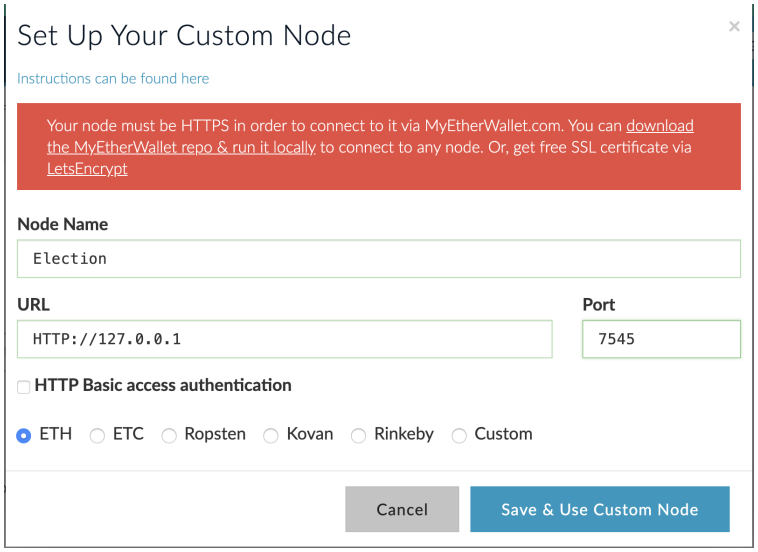


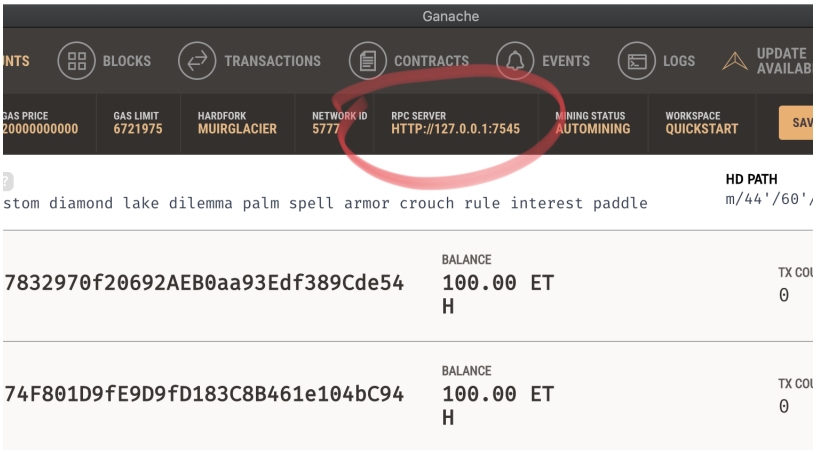
  ○

- Connect it to Ganache
    i. Click on the top-right drop-down menu “Network ETH”
        1. Then add Custom network/ node



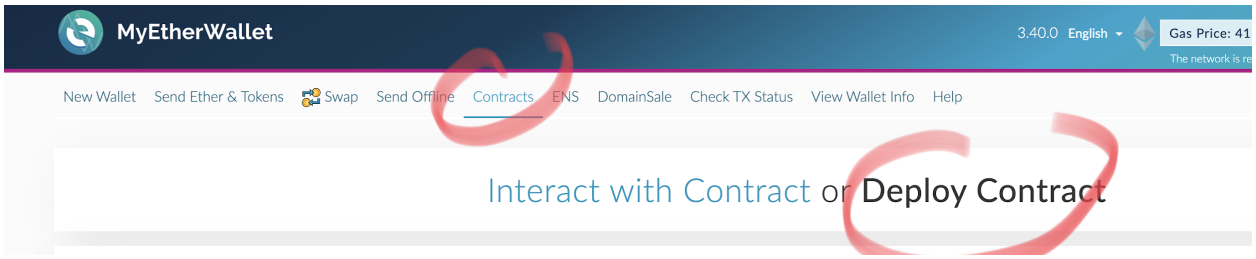
    ii. Fill in the information, URL and Post can be found in Ganache



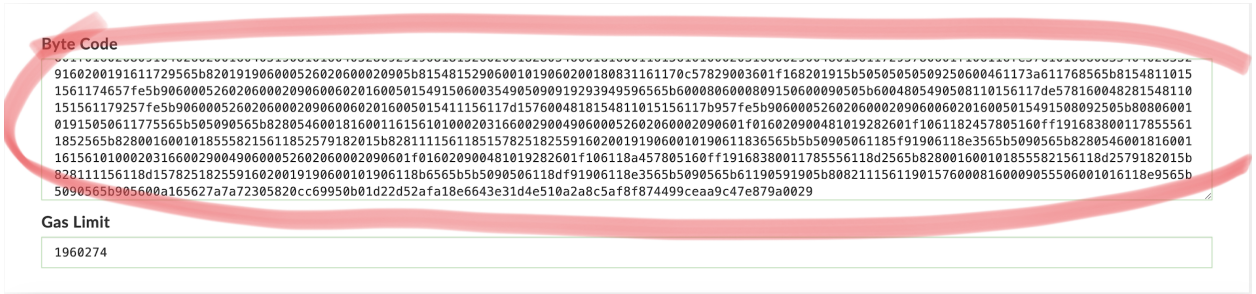


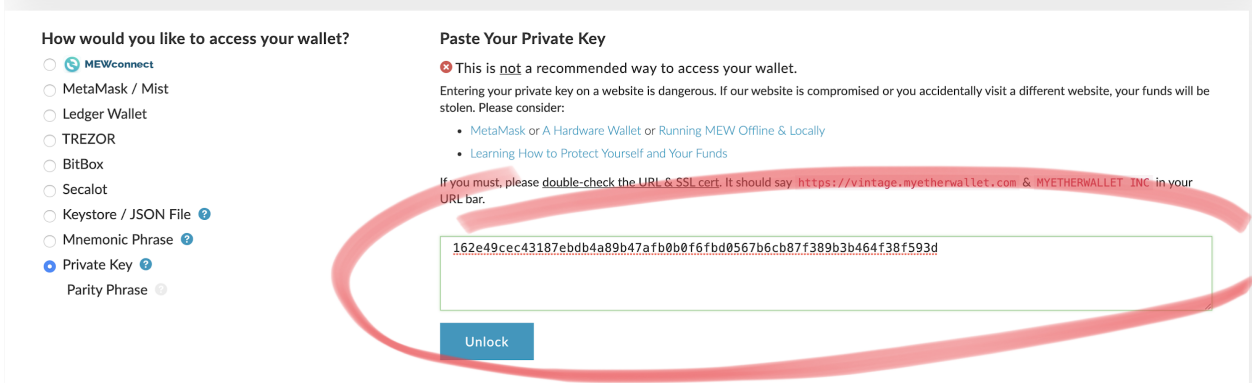
4. “Deploy the contract”

- go to Contracts Tabs and then Deploy contract



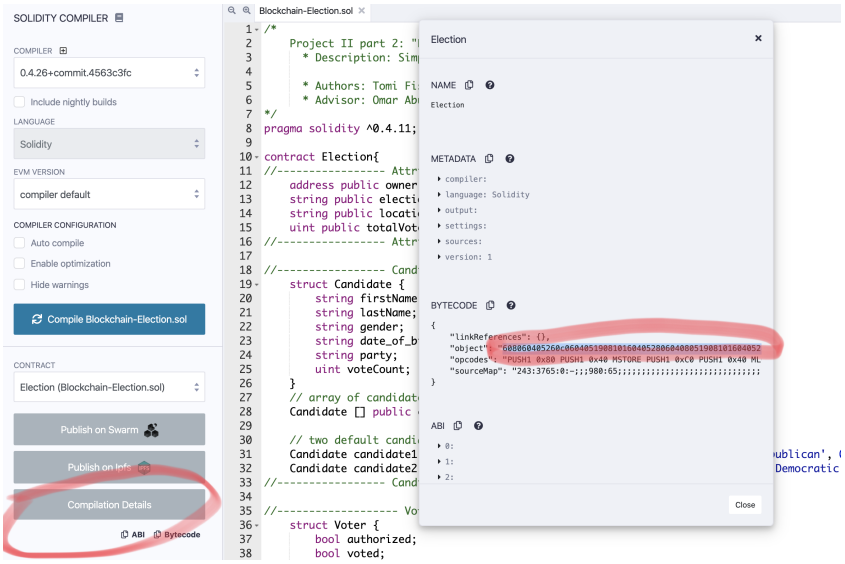
- To Deploy a contract, Fill in the information for Byte code and Private key, Gas Limit is automatically filled in:



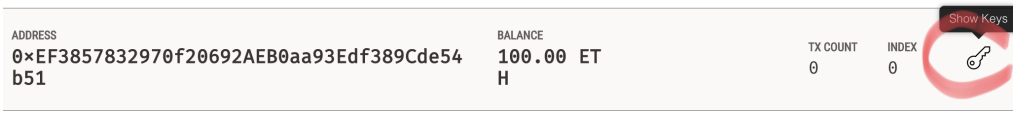


-

- Byte code - can be found in solidity -> compilation detail, note: copy only the data from “object”

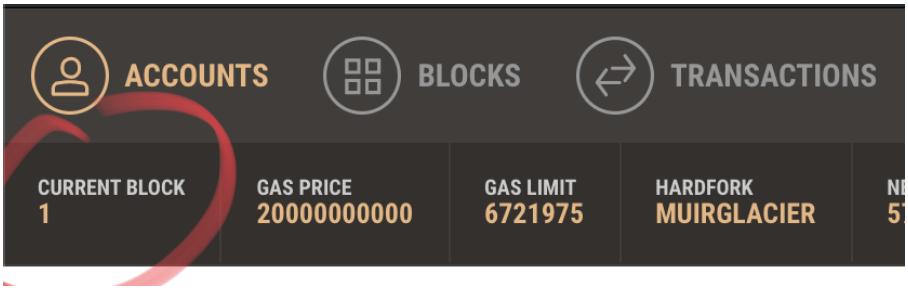


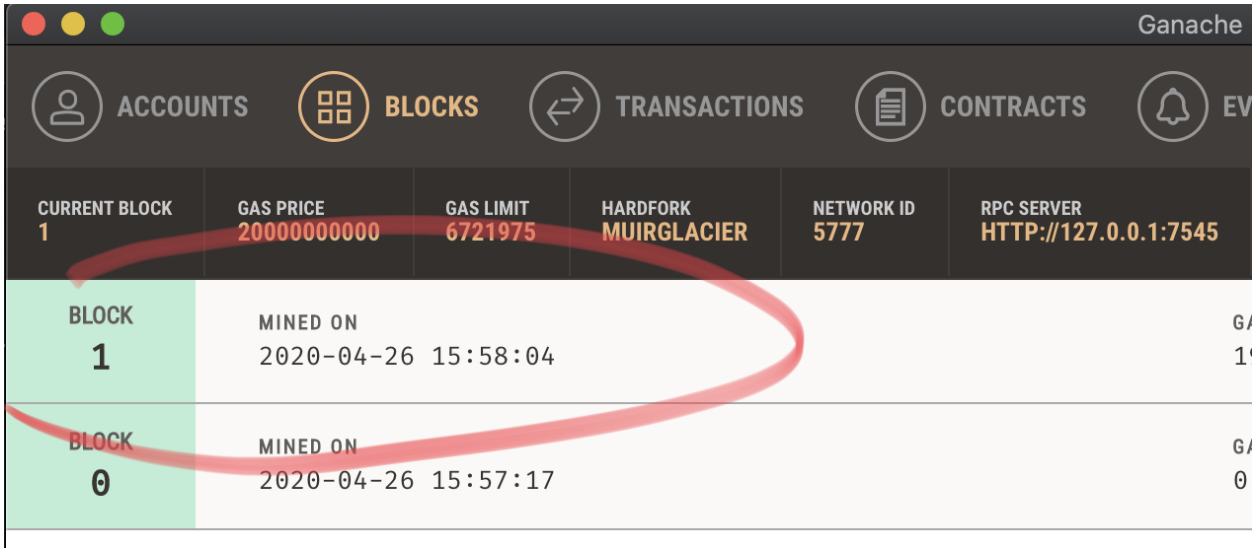
- Private key - can be found in ganache, choose an address, click on the key symbol and copy the private key



- Click unlock and then sign Transaction and Deploy contract
-

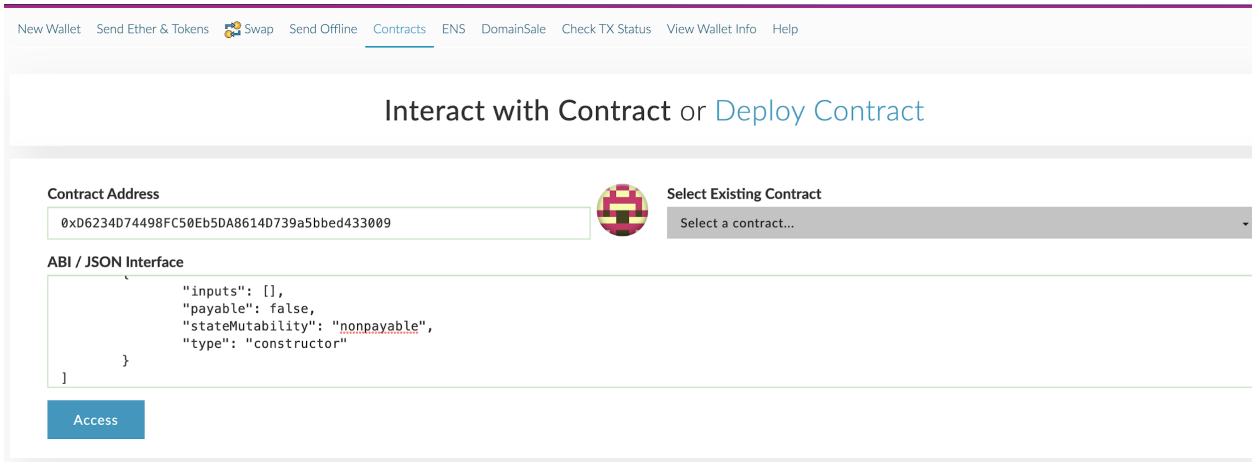
- Result in Ganache:
    - A block is mined - this means that the smart contract is created





5. To “Interact with Contract”

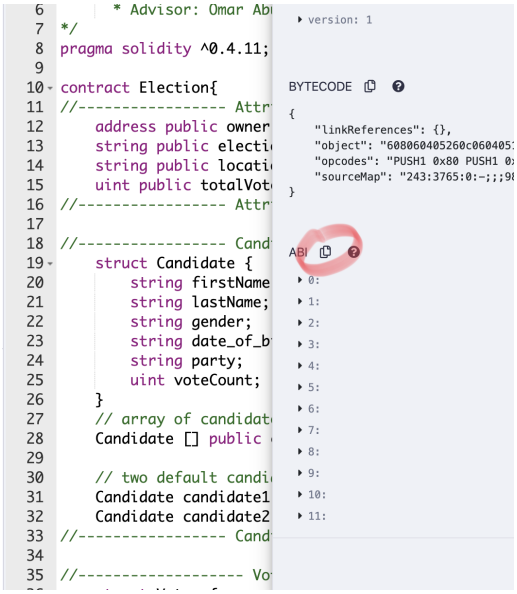
- In myEther, go to Interact with Contract and fill in the information for Contract address and ABI



-

- Contract address - can be found in Ganache -> Transactions -> click on the first block we just mined or created, copy the address from “created contract address”

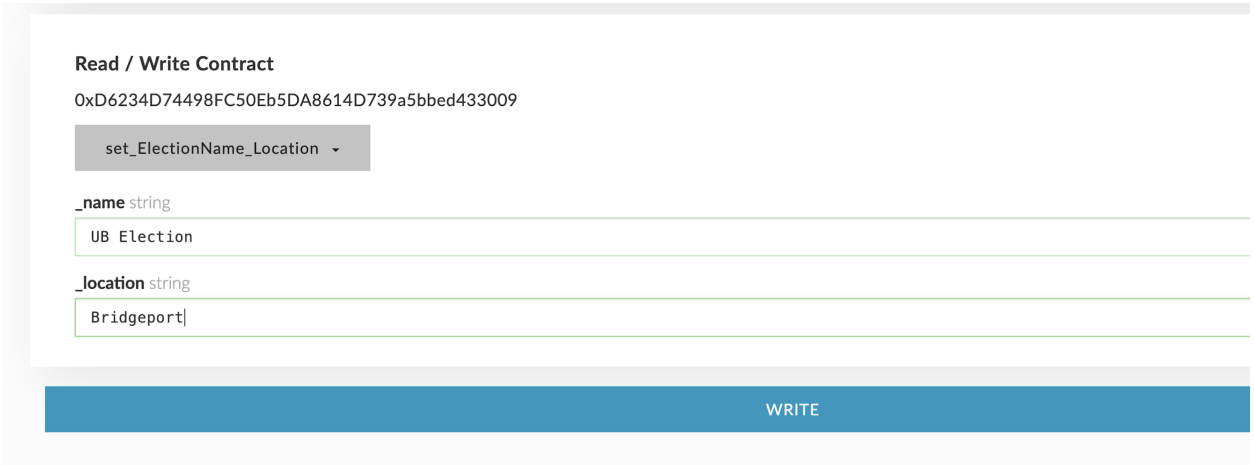
- ABI - can be found in Solidity -> Compilation details -> ABI -> click on this to copy the value



- Finally, click on Access - now we can **interact** with the contract

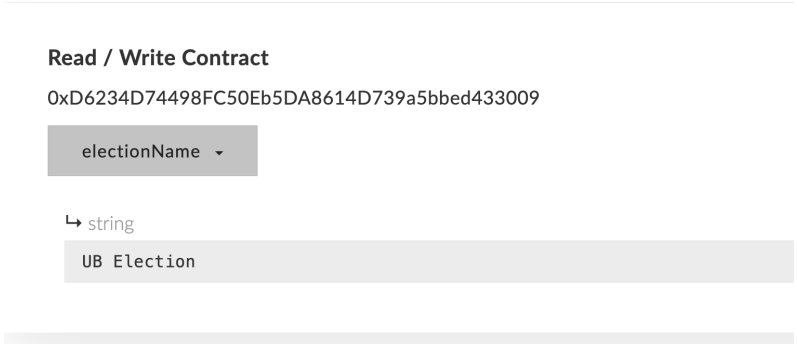
6. Interacting with Contract

- Setting election name and location - using **set_ElectionName_Location**



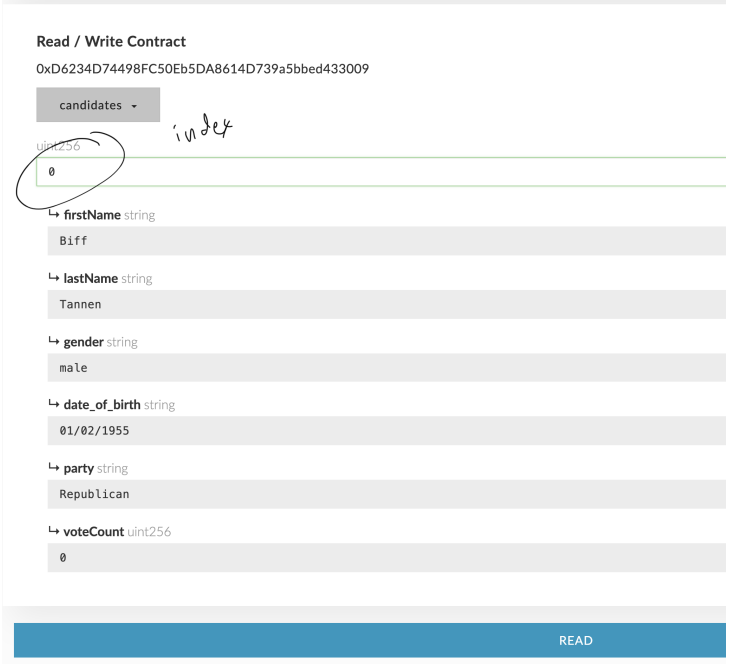
  -

- Check result:

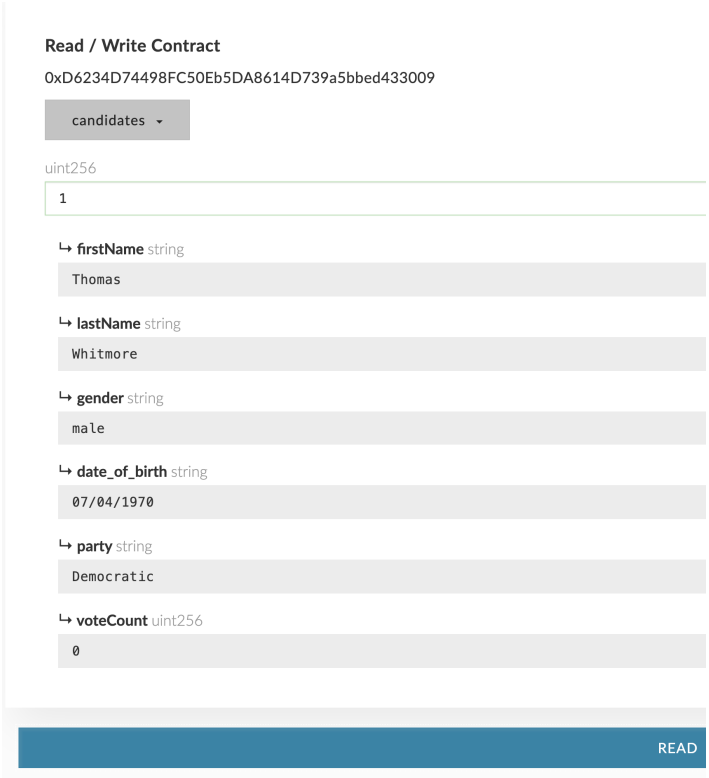


- Get candidates information using **candidates - note:** candidates are stored in an array, access them by inputting the **index**, meaning the first candidate is at index 0 and so on. Also when the contract is created, there will be two default candidates this is due to the constructor in the code
  -

- **Index 0:**

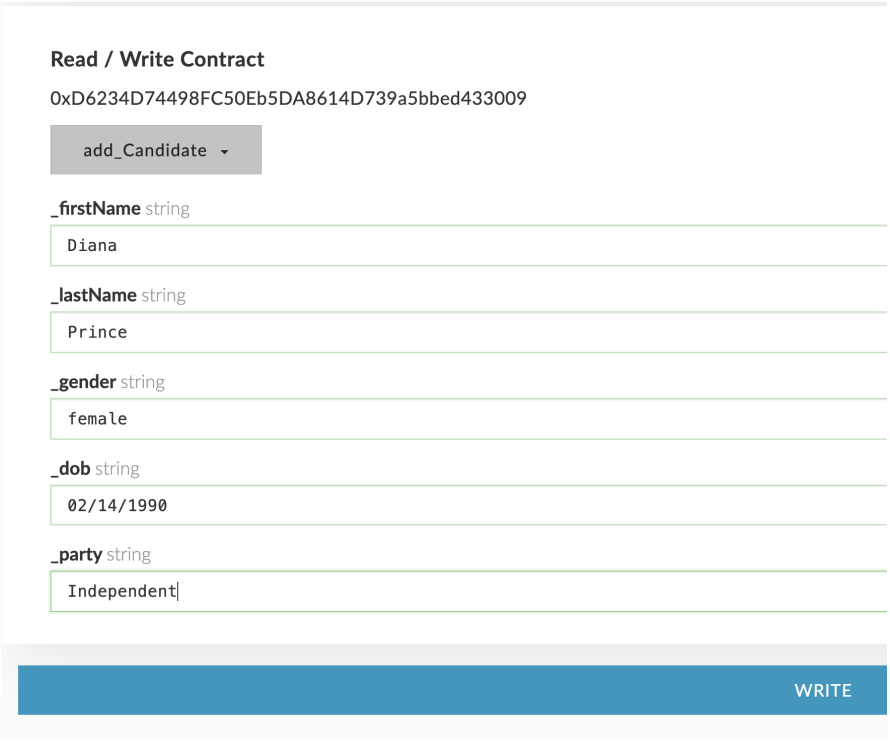


- **Index 1:**

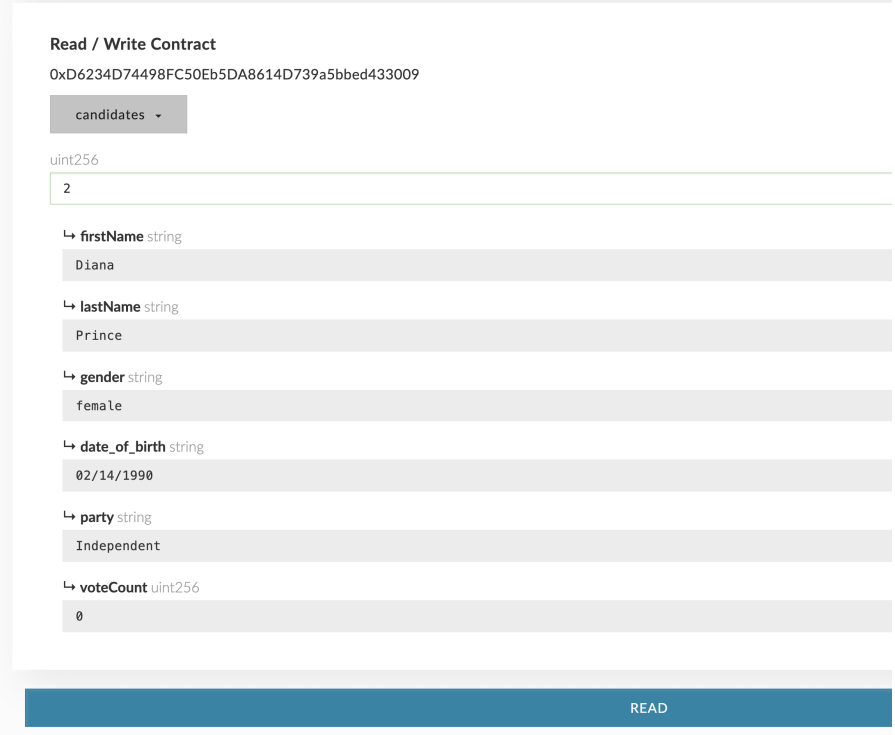


-

- Add Candidate by using **add_Candidate**,

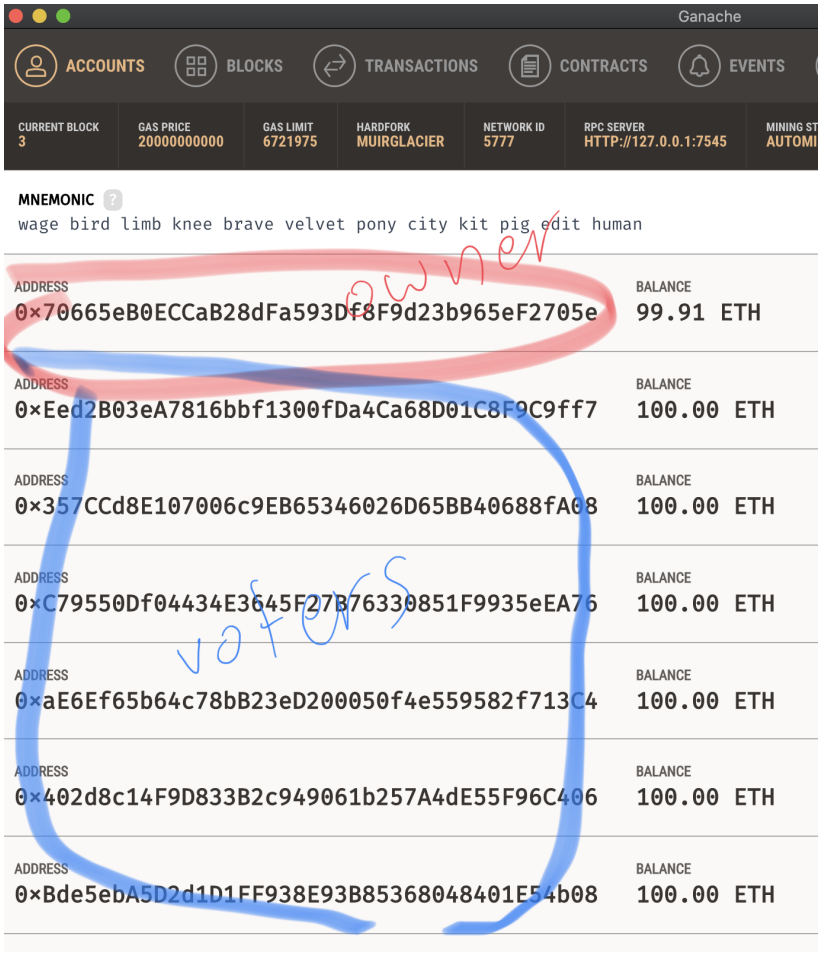


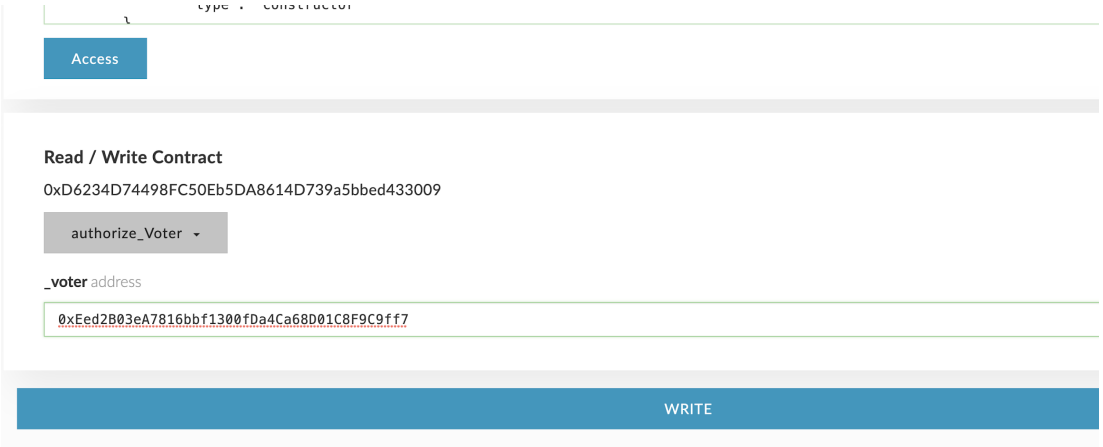
  - Check result by using the **candidates** and inputting the index it was created into, in this case, 2



-

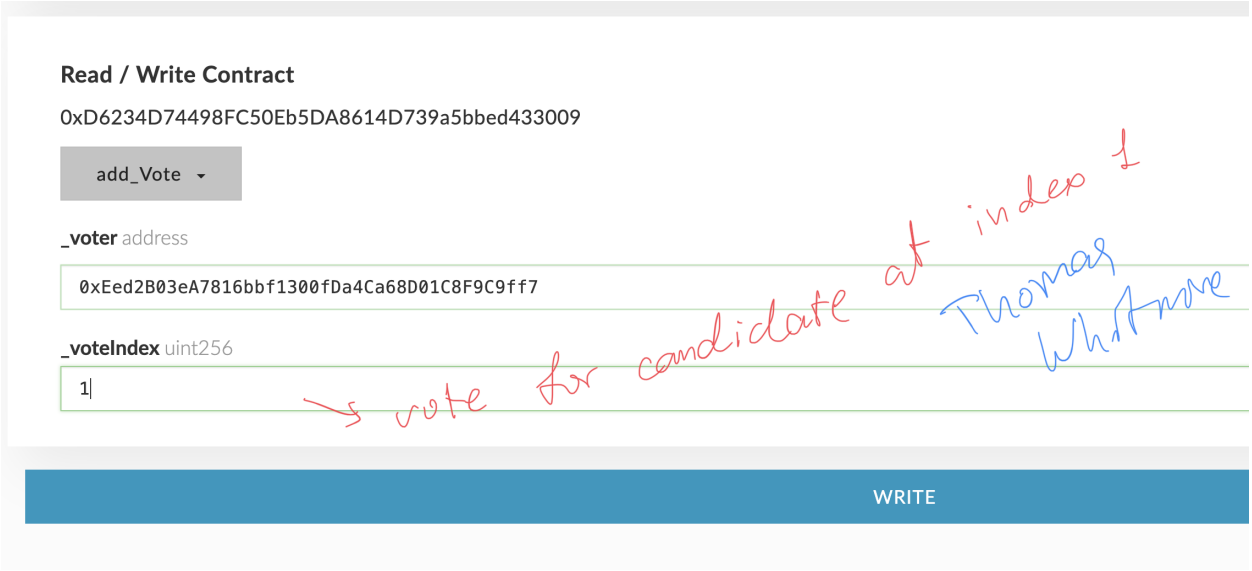
- Authorize a Voter by using **authorized_Voter**: the difference between this and what we did in the app is that in this, the owner of this contract **authorized** the voter whereas in the app there is no authorization we just add the voter into the system or list to able to vote.
    - Assume the **addresses** in ganache is the voter



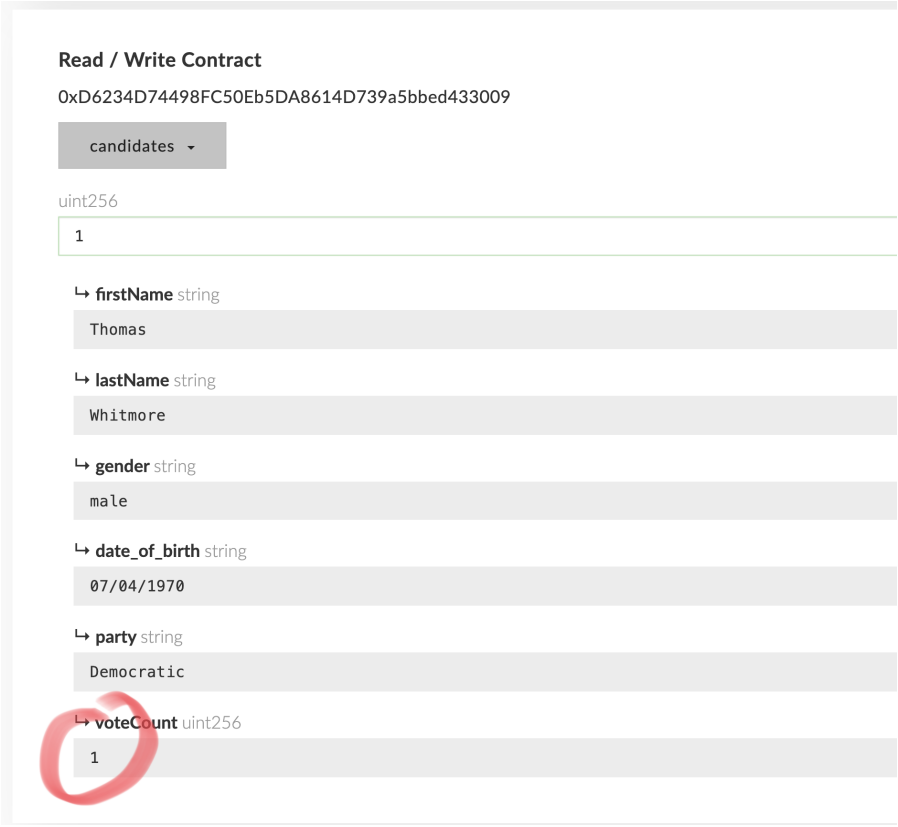


-

- Add Vote by using **add_Vote -** fill in the address we just authorized and input the candidate index - if voter attempted to vote again or is not authorized - there will be an **error**

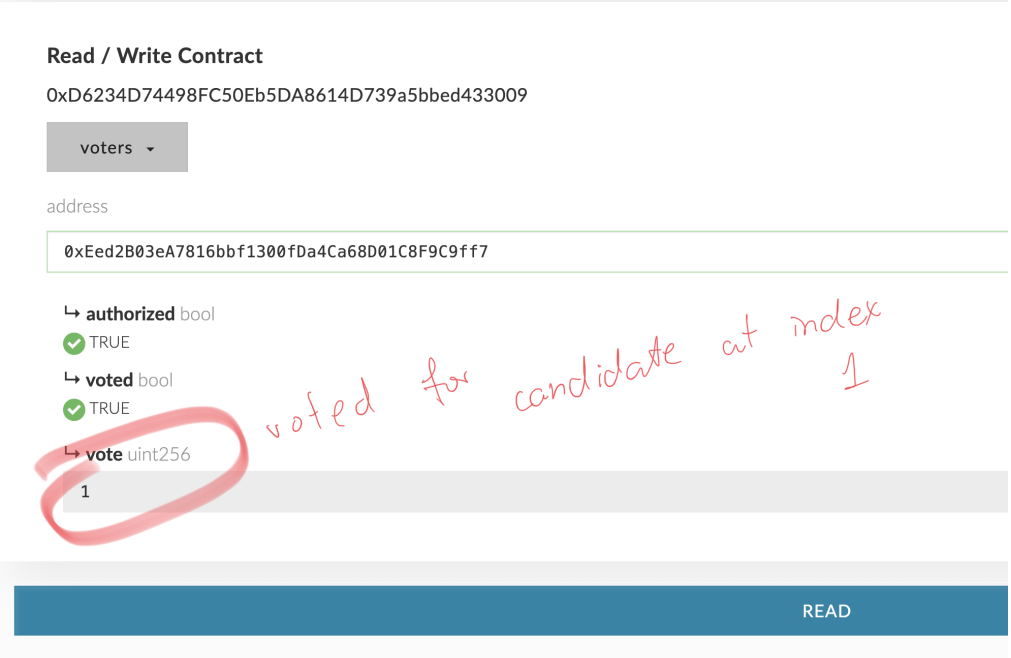


  - Check result

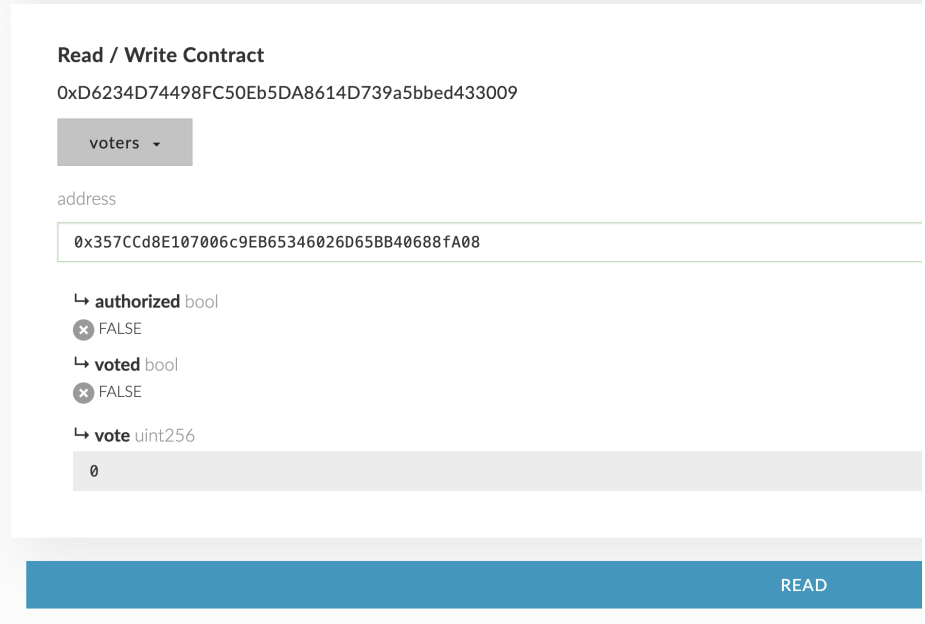


-

- Get Voter information using **voters-** put in the address we just authorized and used to vote:



A voter that has not authorized and voted yet:



-

- Get the winner using **winnerName**

Read / Write Contract

0xD6234D74498FC50Eb5DA8614D739a5bbed433009

winnerName

↳ **_firstName** string

```
Thomas
```

↳ **_lastName** string

```
Whitmore
```

↳ **_gender** string

```
male
```

↳ **_dob** string

```
07/04/1970
```

↳ **_party** string

```
Democratic
```

votes for this candidate

↳ **_votes** uint256

```
3
```

↳ **_totalVotes** uint256

```
5
```

total votes in this election