Part 2 – Code and Deploy a smart contract on Ethereum testnet “Ropsten”

IMPORTANT : At the end of your exercise push the code of the smart contract on your created github project “Blockchain distributed system & Smart contract” where you create a new repository name “Tutorial 2 : Part 2”.

When you finished to complete the Quest please provide your github account with all files (configurations, smart contract code …) created for this tutorial including this tutorial;

- Public key of your metamask account
- All the transactions ID generated
- Send me 1 ETH on my Ethereum Wallet on ropsten testnet with the metamask account used for this tutorial : 0xddF8CDa9d8A425b5a8FFb64BfC79857bA9b57083

You have to create also a report with all print screens for each following steps.

Pré-requis :

Code du Smart contract à déployer : https://github.com/cozcan/KamalRajniElection

Metamask un wallet intégrer dans votre navigateur Chrome .metamask.io et détenir des Ethers sur Ropsten

Remix : Remix - Solidity IDE disponible via votre navigateur remix.ethereum.org

Ropsten Test Network : Environment de test

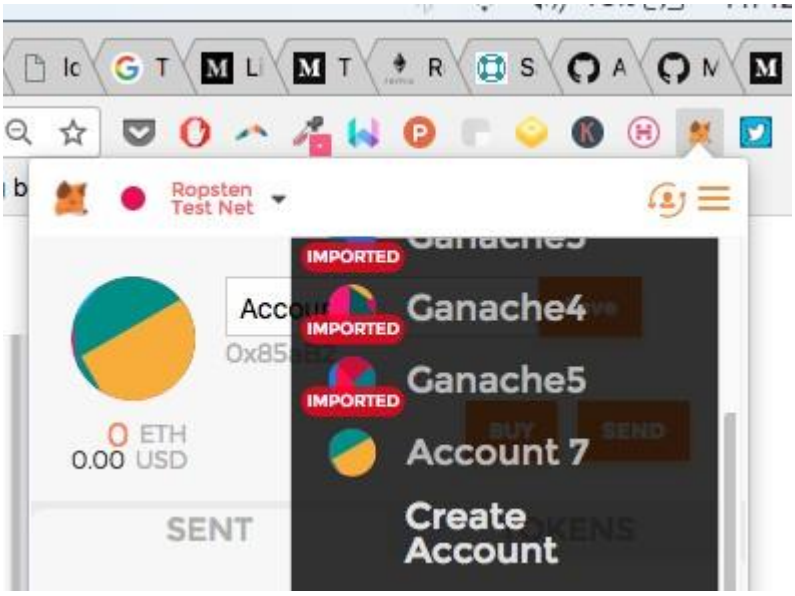
Stage 1: Setup the Environment

Copy the election.sol (smart contract) and paste it into the remix IDE.

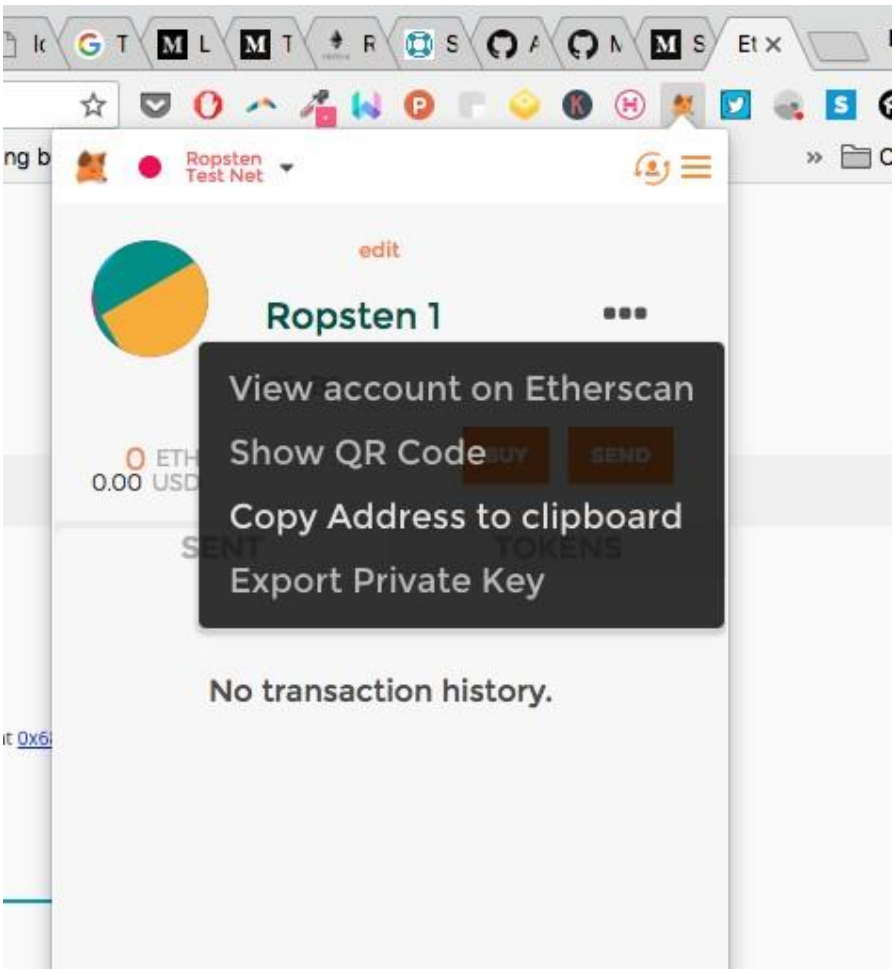
To deploy the contract, we need an account and with some ether on the Rospten test net.

- Meta mask!
- Select Rospten Test net

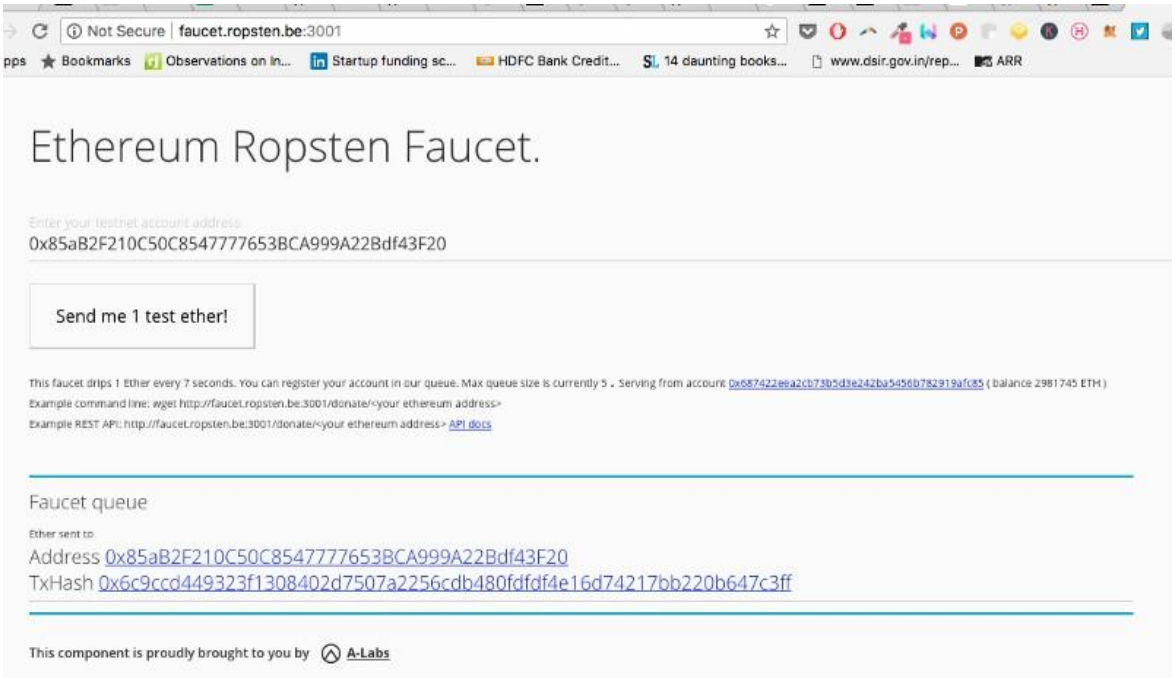
- Create an Account



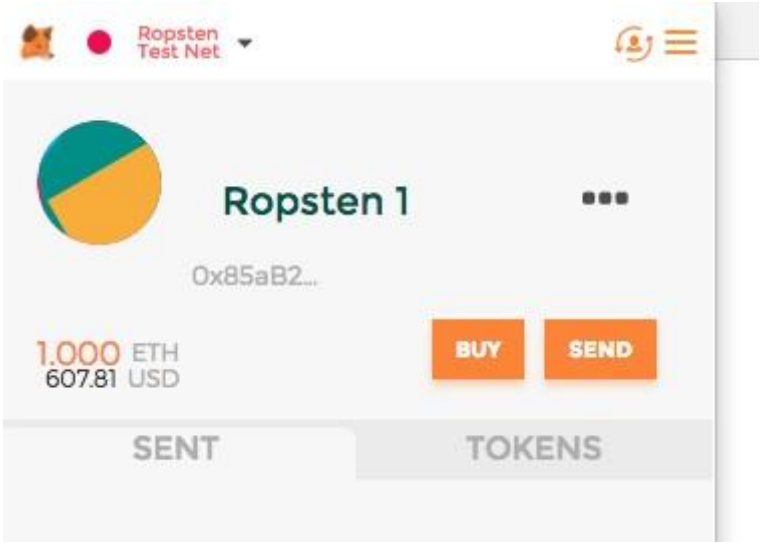
- Copy Account Address



- Request Free ether from here or here. Paste the copied account address in text box and click on send me 1 test ether.



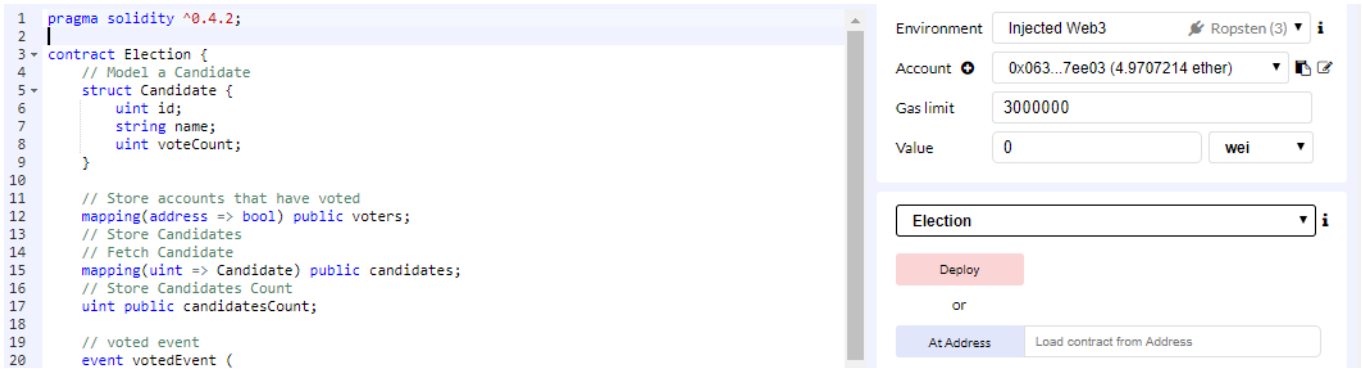
- Ogle at your free ether :P



So now, we have the contract pasted in remix IDE, we have connected meta mask to the ropsten testnet with an account that has some ether.

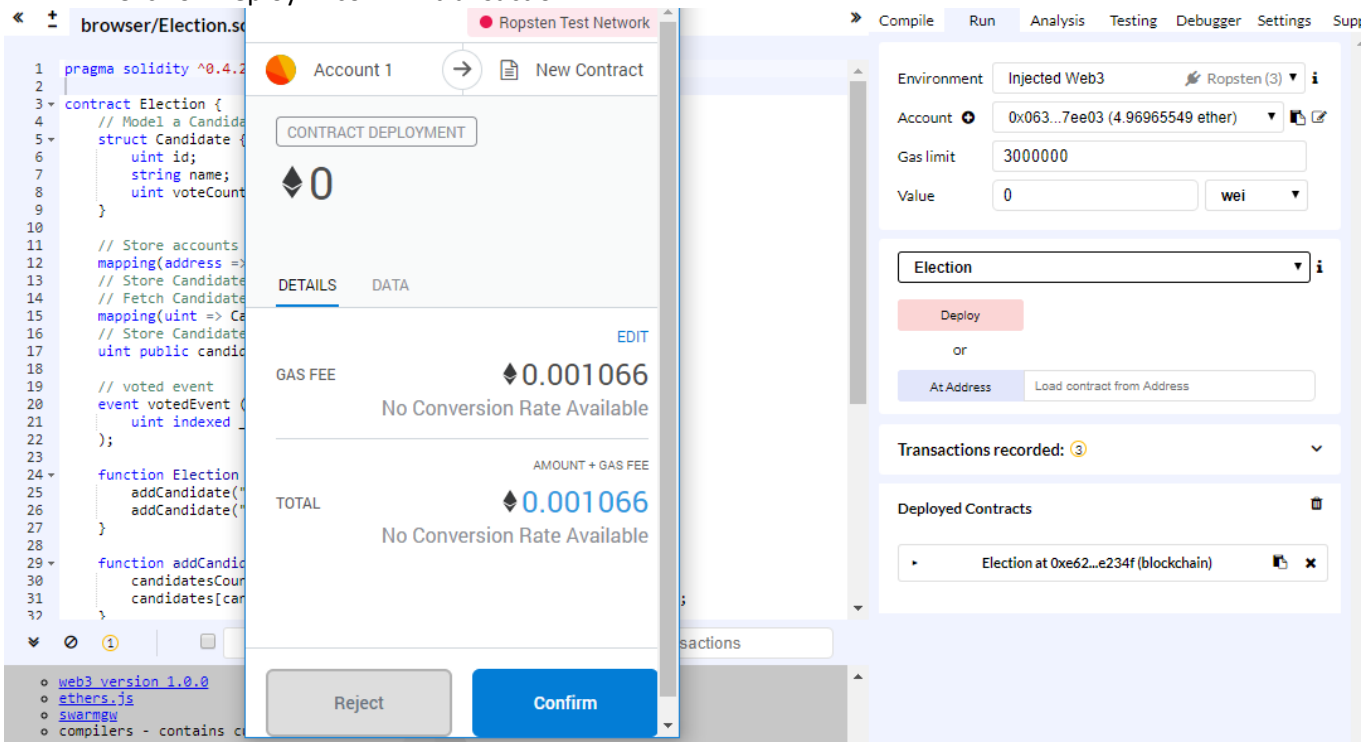
Stage 2: Deploy the contract

- click on Run. Select Injected Web 3 Ropsten under environment and the account in Metamask is shown here under Account with balance ether as well.

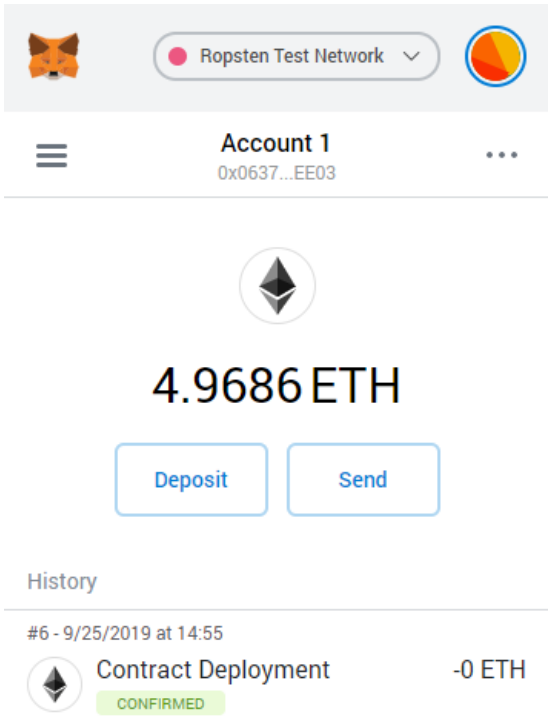


```
pragma solidity ^0.4.2;

contract Election {
    // Model a Candidate
    struct Candidate {
        uint id;
        string name;
        uint voteCount;
    }

    // Store accounts that have voted
    mapping(address => bool) public voters;
    // Store Candidates
    // Fetch Candidate
    mapping(uint => Candidate) public candidates;
    // Store Candidates Count
    uint public candidatesCount;

    // voted event
    event votedEvent (
```

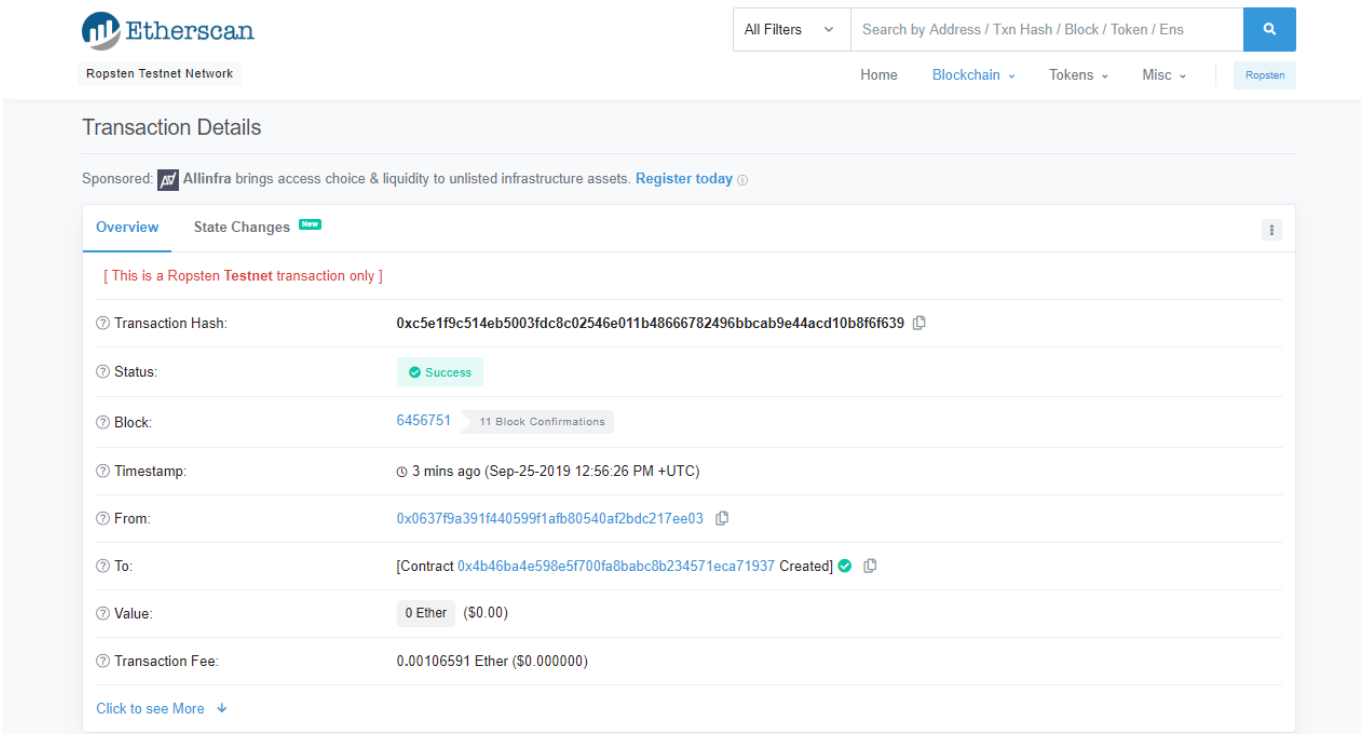
- Click on Deploy -> confirm transaction



- Click on Metamask extension!

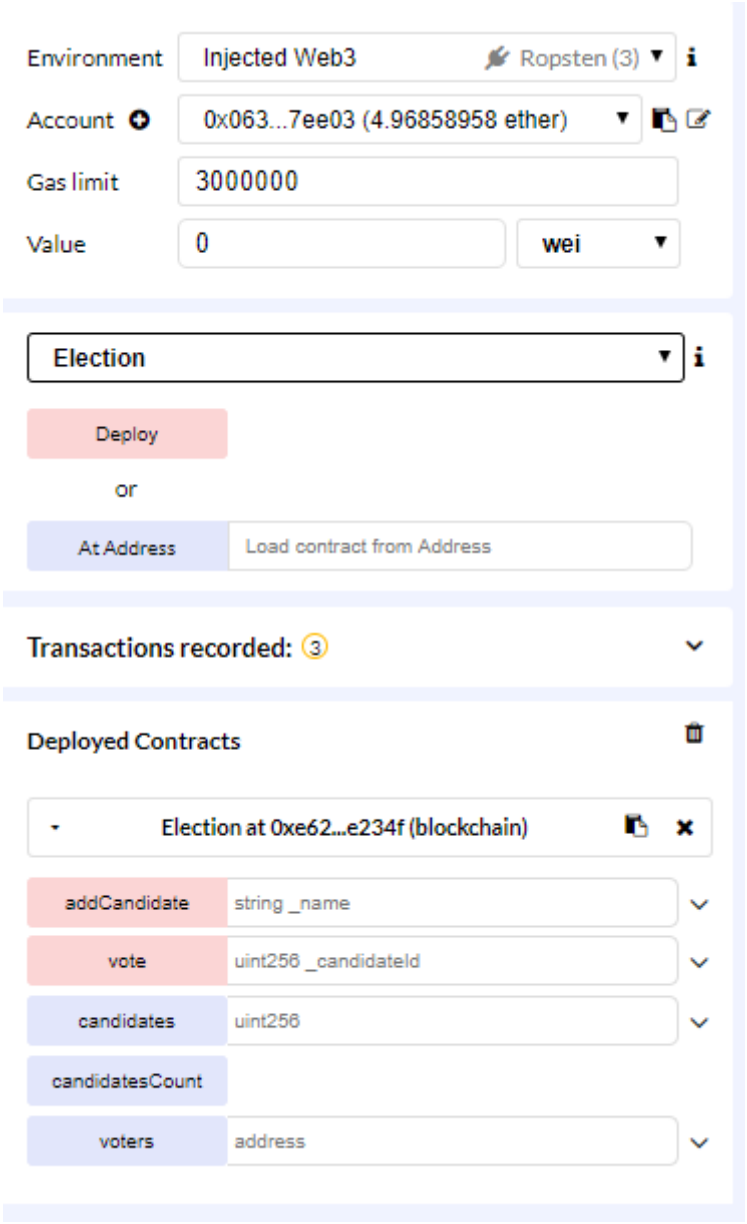


- Click on the contract deployment — it should open up the etherscan page to look at our transaction details. Check the status is successful



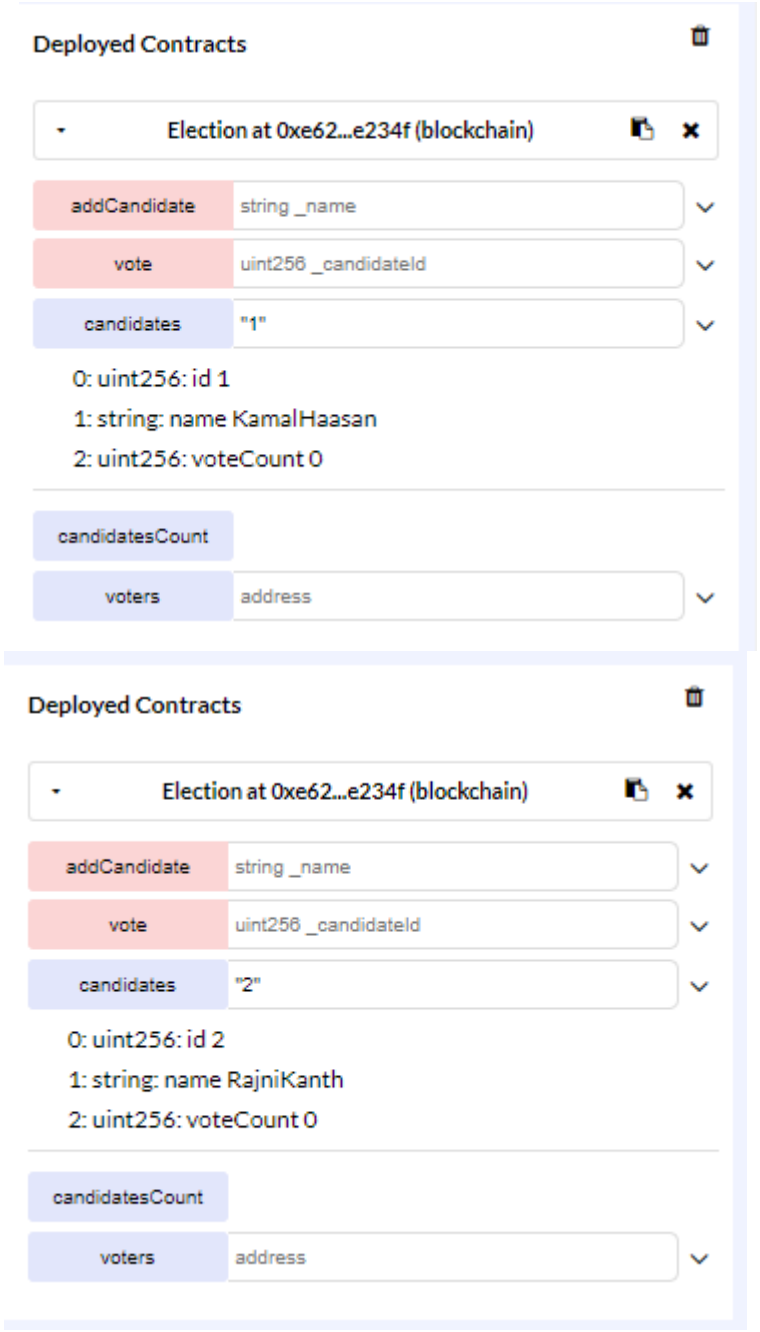
congrats, you have successfully created a smart contract

Stage 3: Interact with the contract



- **Pull / Access data from contract— free — shaded "blue"**

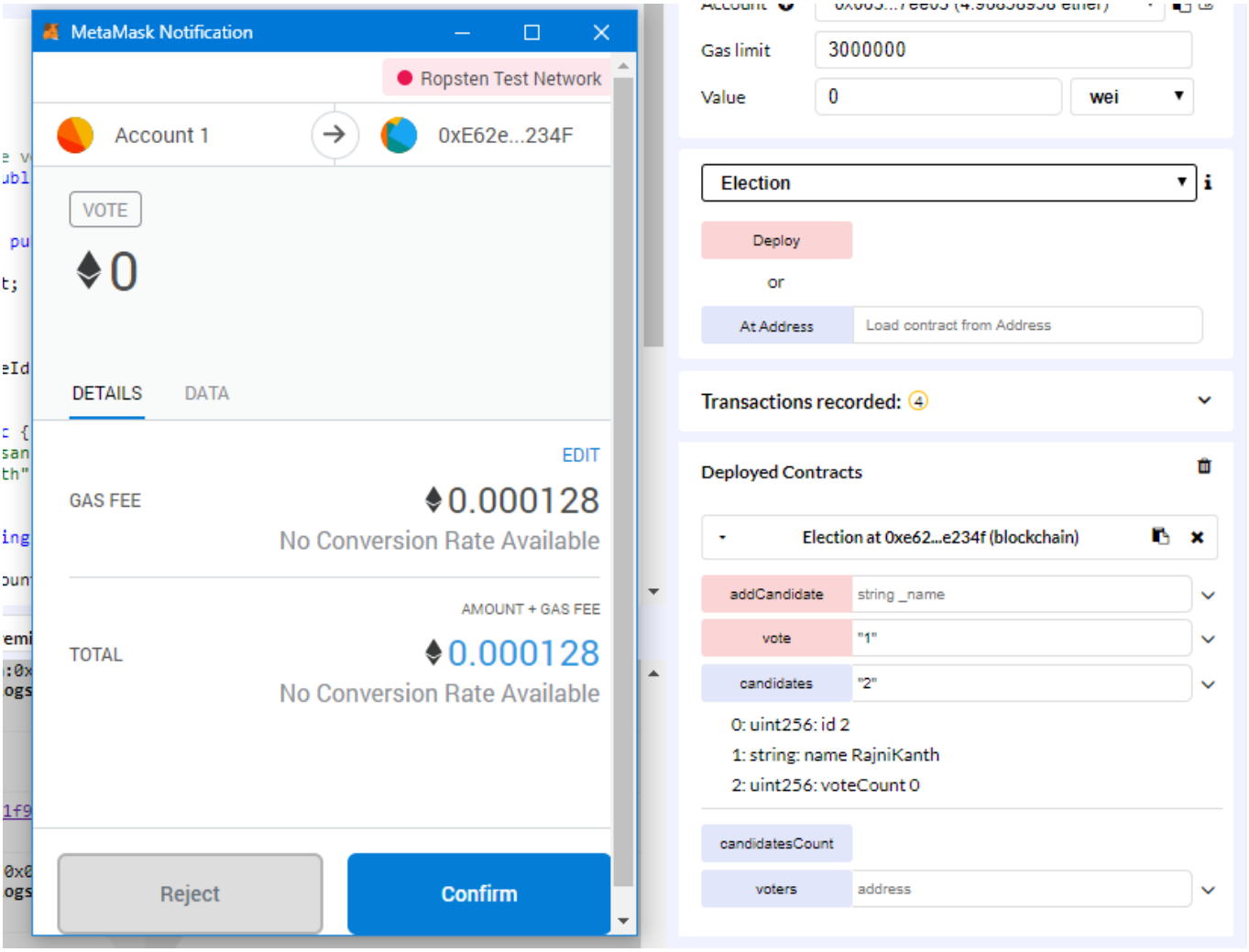
If you add 1 (candidate ID) near the candidates field ..you should get the details of candidate ID 1. And 2 would give you the second candidate details.



remember pulling data is free on ethereum blockchain

- **Vote / add data to the chain!**

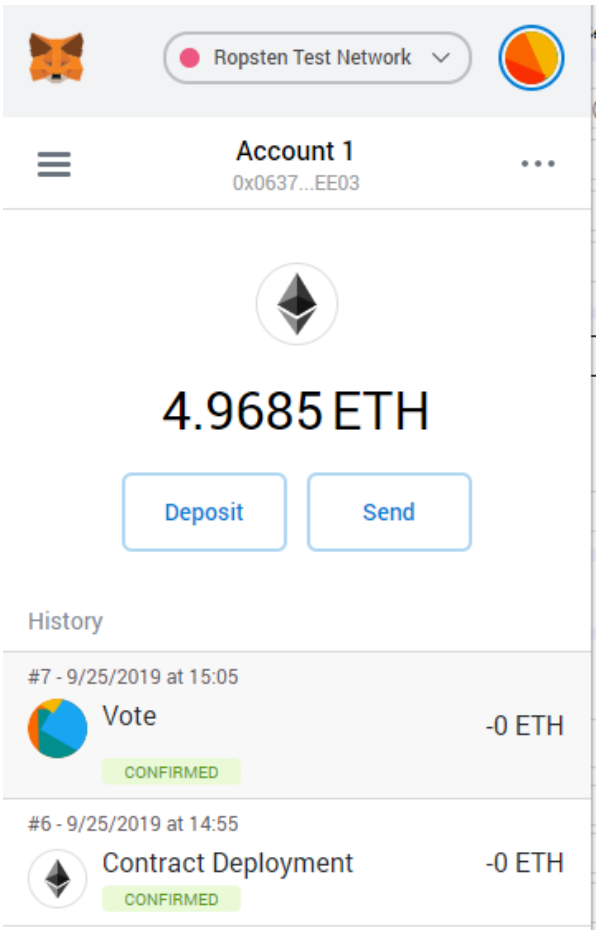
Now that we know who is whom. Let's try and Vote.



notice the 1 next to vote

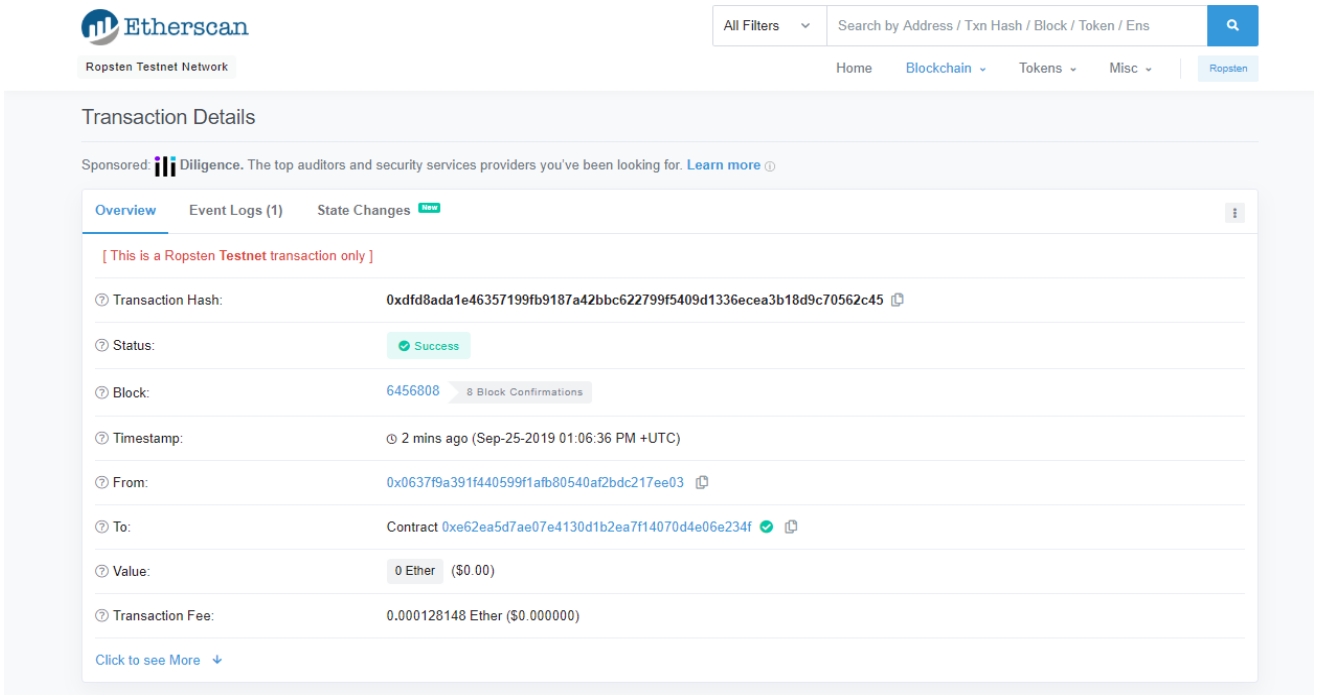
```
transact to Election.vote pending ...

https://ropsten.etherscan.io/tx/0xdfd8ada1e46357199fb9187a42bbc622799f5409d1336ecea3b18d9c70562c45
```

IDE shows my transaction is pending

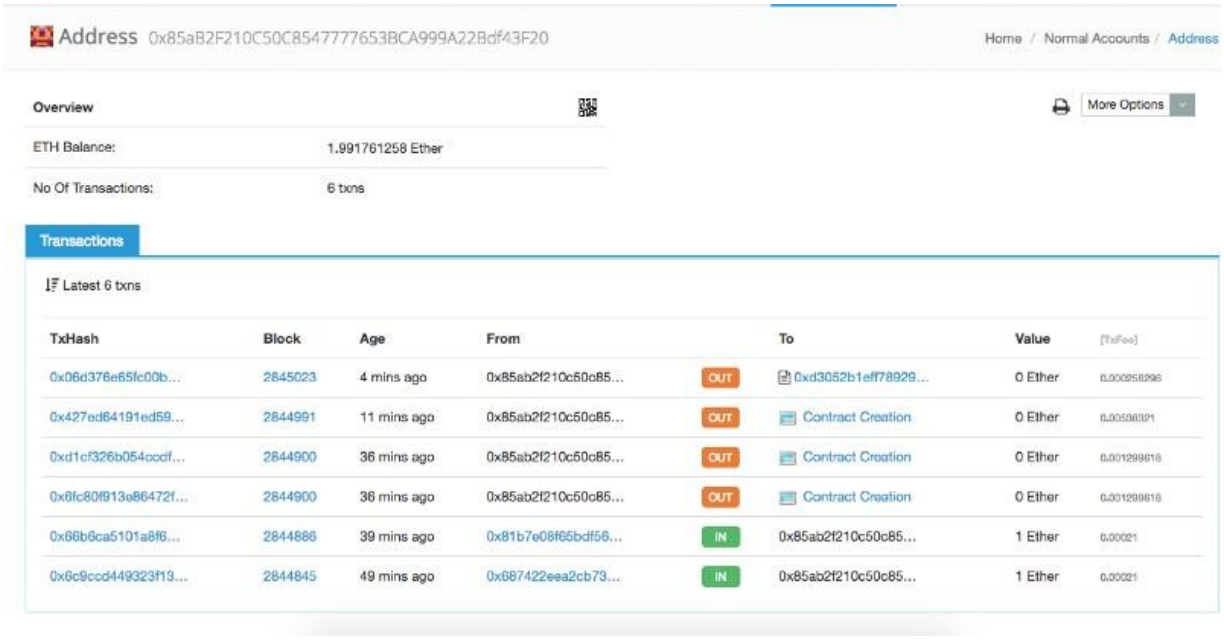


after few seconds, the Metamask extension shows me the transaction details. If I click on it, it should take me to ether scan.

Transaction successful!



If I copy my address from metamask and paste it into etherscan I can see all the transactions within etherscan



The Green INs are the ethers that have come to me from the faucets.
The Orange OUTs are my spends. I have created the contract thrice and wrote on it once.

If I click on the first line item the vote transaction, this is what we see in etherscan.