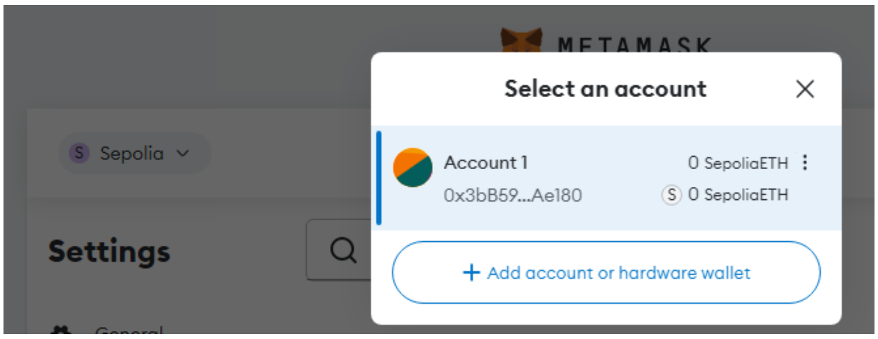
Lab 4: Ethereum hands on 1

Metamask

Sepolia (test)

Step 1: We created a Metamask account and added an extension on Google Chrome



METAMASK

Sepolia

Settings

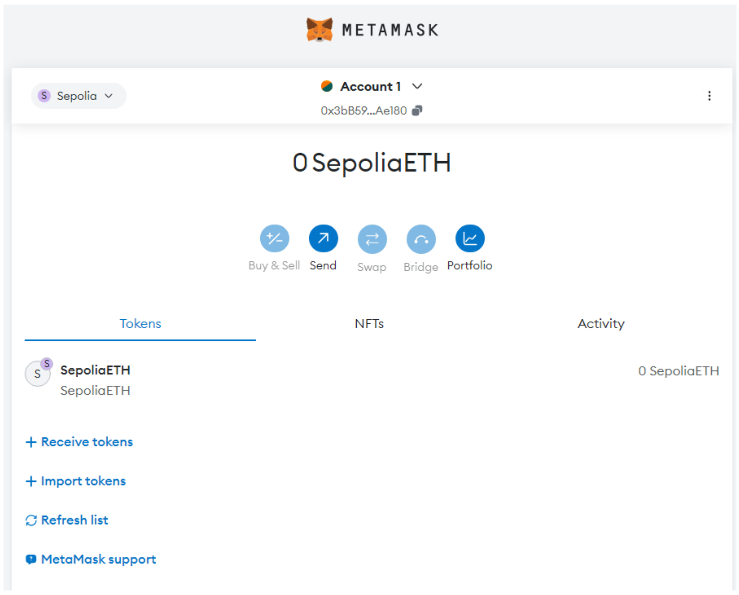
Select an account

Account 1 0 SepoliaETH
0x3bB59...Ae180 0 SepoliaETH

+ Add account or hardware wallet

We are associated with the below account number for our wallet in Metamask. We add SepoliaETH network

Account 1: 0x3bB59688853227ff584f2D25Fd0bD75a9f3Ae180



METAMASK

Sepolia

Account 1
0x3bB59...Ae180

0 SepoliaETH

SepoliaETH
SepoliaETH 0 SepoliaETH

+ Receive tokens

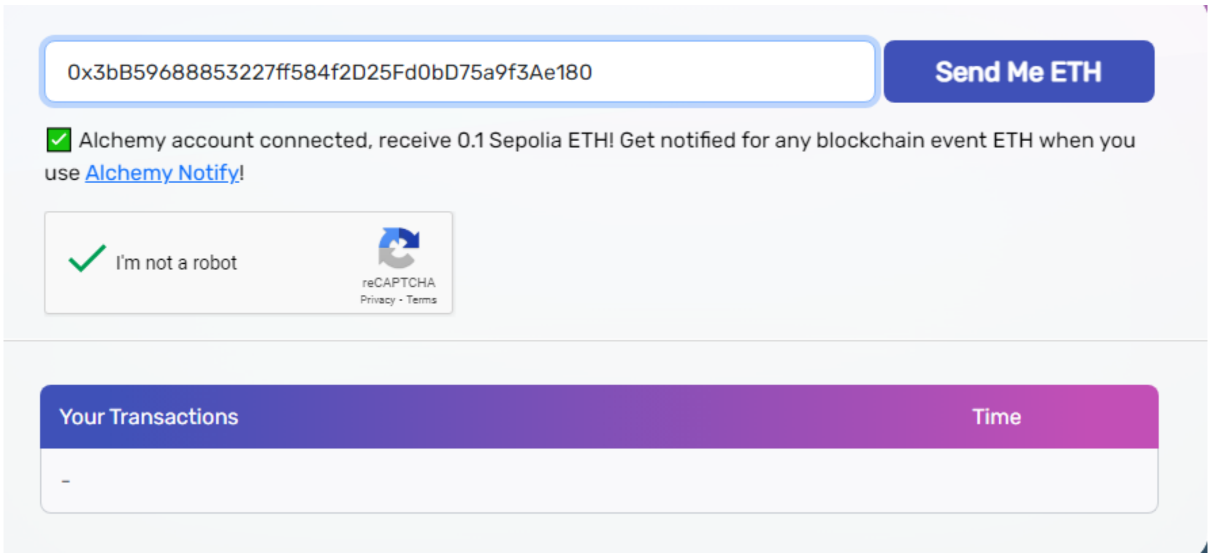
+ Import tokens

Refresh list

MetaMask support

Step 2

We attempt to use SepoliaETH Alchemy test faucet to send eth – however unable to do so as needing to have actual 0.001eth


0x3bB59688853227ff584f2D25Fd0bD75a9f3Ae180

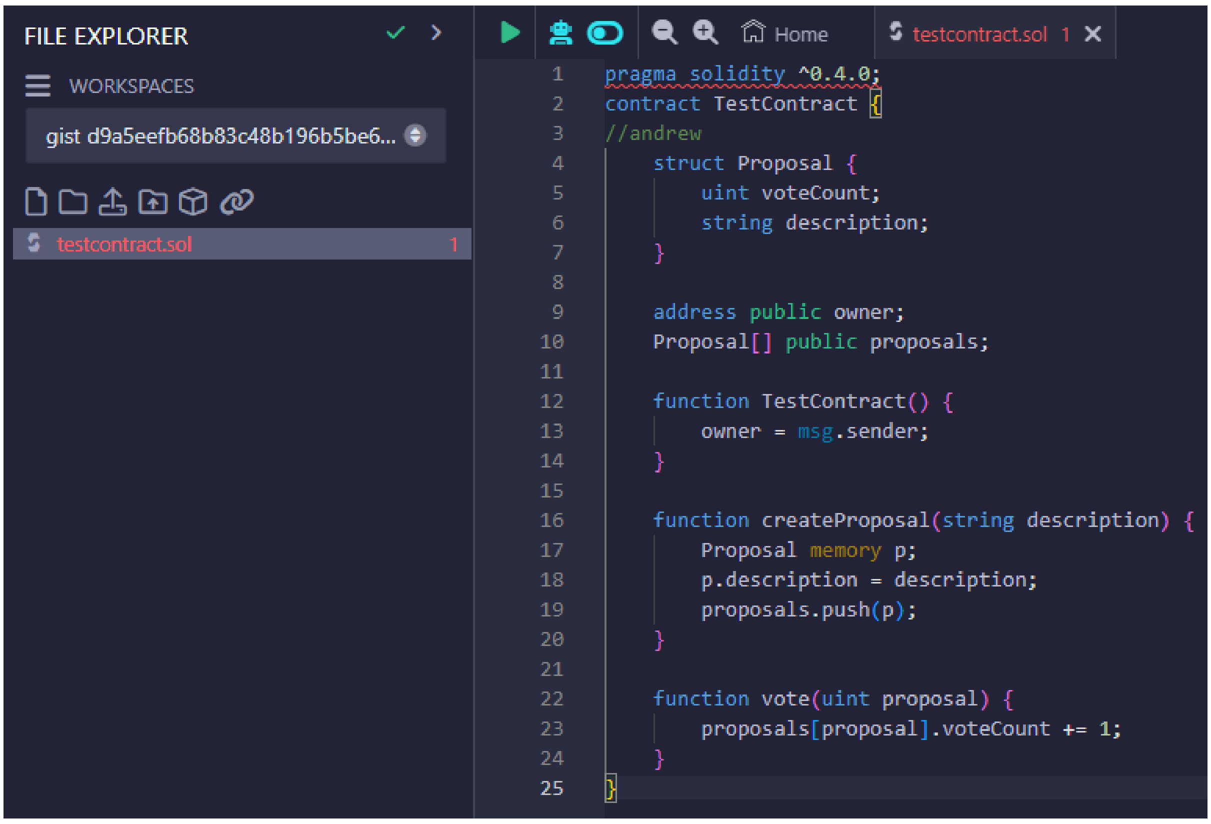
✅ Alchemy account connected, receive 0.1 Sepolia ETH! Get notified for any blockchain event ETH when you use Alchemy Notify!

Your Transactions	Time
-	

Step 3:

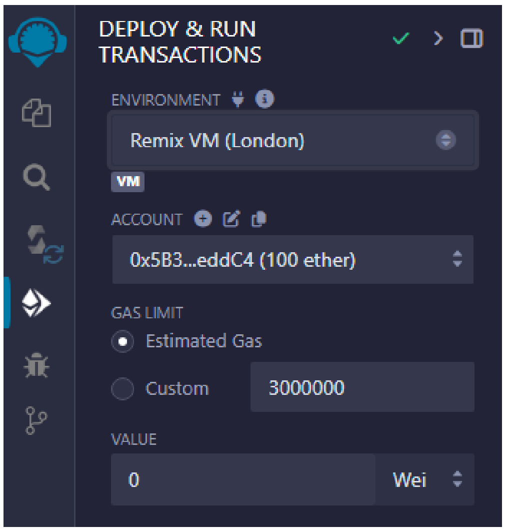
We load the contract in the remix IDE

http://remix.ethereum.org/#gist=d9a5eefb68b83c48b196b5be65f1be54&lang=en&optimize=false&runs=200&evmVersion=null&version=soljson-v0.4.0+commit.acd334c9.js&language=Solidity



```
pragma solidity ^0.4.0;
contract TestContract {
//andrew
    struct Proposal {
        uint voteCount;
        string description;
    }

    address public owner;
    Proposal[] public proposals;

    function TestContract() {
        owner = msg.sender;
    }

    function createProposal(string description) {
        Proposal memory p;
        p.description = description;
        proposals.push(p);
    }

    function vote(uint proposal) {
        proposals[proposal].voteCount += 1;
    }
}
```

We had to compile the contract before clicking on Deploy and run transactions


DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Remix VM (London)

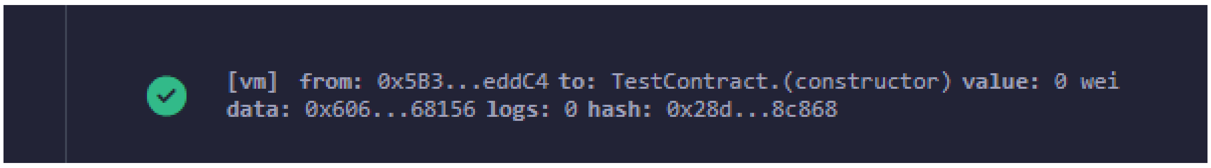
ACCOUNT: 0x5B3...eddC4 (100 ether)

GAS LIMIT: Estimated Gas

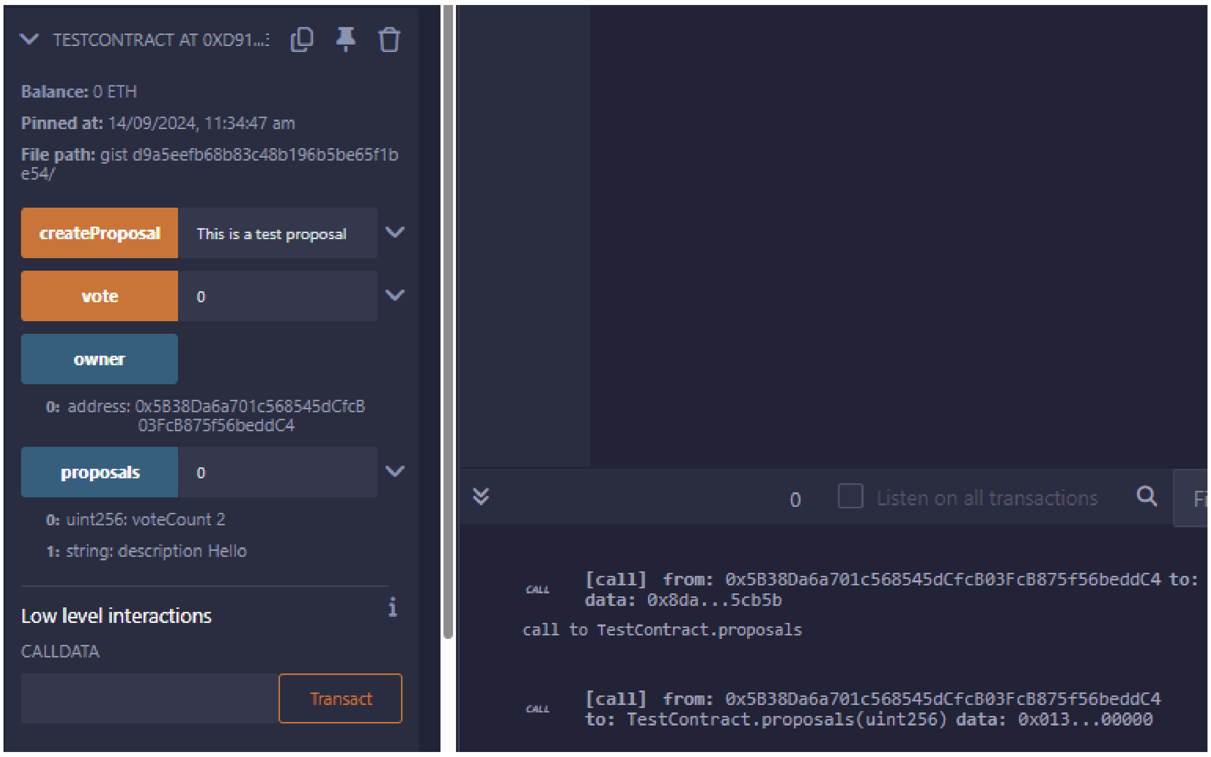
Custom: 3000000

VALUE: 0 Wei

We click deploy on the IDE pane


```
[vm] from: 0x5B3...eddC4 to: TestContract.(constructor) value: 0 wei
data: 0x606...68156 logs: 0 hash: 0x28d...8c868
```

We begin the query proposals field by index


TESTCONTRACT AT 0XD91...

Balance: 0 ETH

Pinned at: 14/09/2024, 11:34:47 am

File path: gist d9a5eefb68b83c48b196b5be65f1be54/

createProposal: This is a test proposal

vote: 0

owner

0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

proposals: 0

0: uint256: voteCount 2

1: string: description Hello

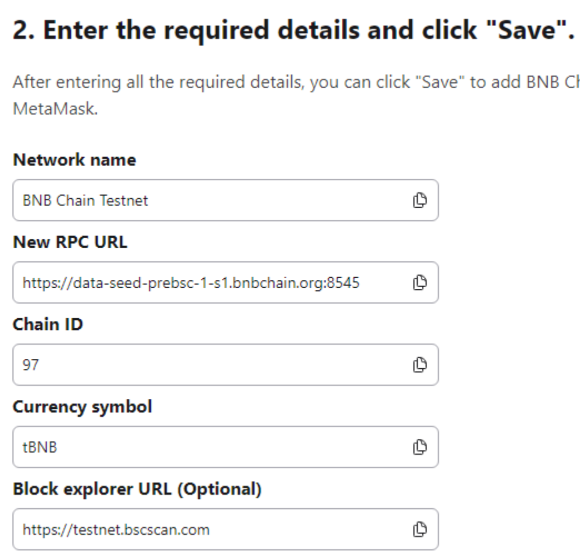
Low level interactions

CALLDATA

Transact

```
0   Listen on all transactions

[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to:
data: 0x8da...5cb5b
call to TestContract.proposals

[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to: TestContract.proposals(uint256) data: 0x013...00000
```

We find another testnet that can send BNB testnet. So using Metamask widget we add a new network BNB Chain Testnet and add the required details below



2. Enter the required details and click "Save".

After entering all the required details, you can click "Save" to add BNB Cl
MetaMask.

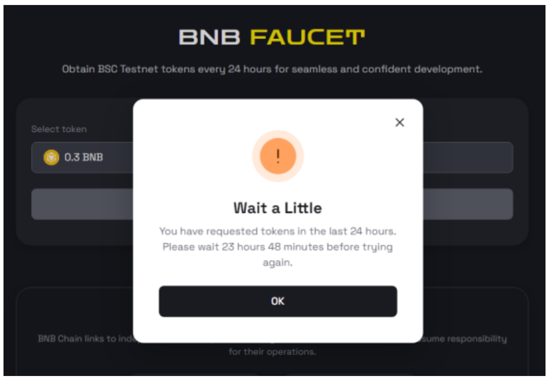
Network name	BNB Chain Testnet
New RPC URL	https://data-seed-prebsc-1-s1.bnbchain.org:8545
Chain ID	97
Currency symbol	tBNB
Block explorer URL (Optional)	https://testnet.bscscan.com

We visit the BNB Faucet https://www.bnbchain.org/en/testnet-faucet and are able to send 0.3 BNB tokens to our MetaMask wallet with the BNB Chain Testnet network that has been added.

(I copied the wallet address: 0x3bB59688853227ff584f2D25Fd0bD75a9f3Ae180

Added this number to onedrive, and accessed onedrive from mobile device and copied this number.

I access the website via mobile device and entered the wallet address and was successful in sending 0.3TBNB to my Metamask wallet address



BNB FAUCET

Obtain BSC Testnet tokens every 24 hours for seamless and confident development.

Select token

0.3 BNB

Wait a Little

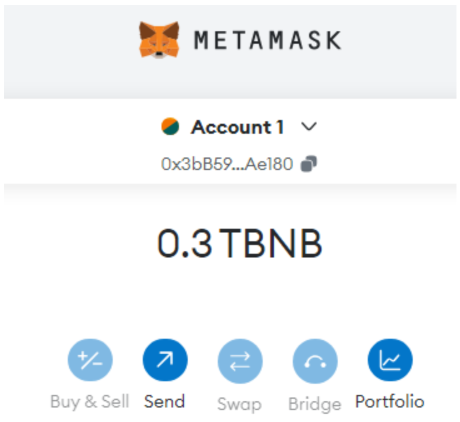
You have requested tokens in the last 24 hours. Please wait 23 hours 48 minutes before trying again.

OK

BNB Chain links to ind ume responsibility for their operations.

Below we see confirmation of the TBNB

chrome-extension://nkbihfbeogaeaoehlefnkodbefgpgknn/home.html#

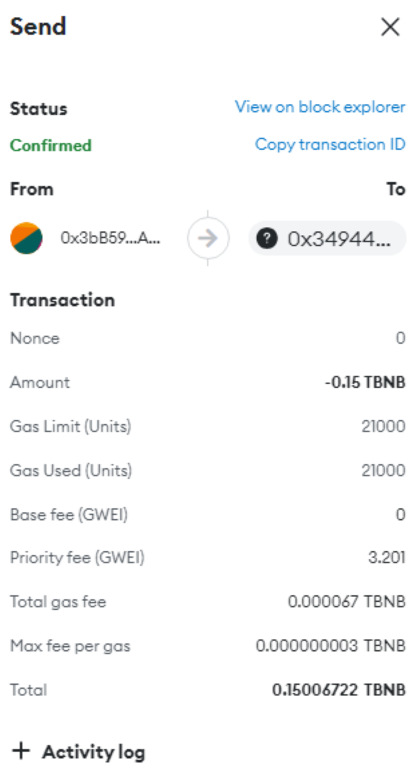

METAMASK

Account 1

0x3bB59...Ae180

0.3 TBNB

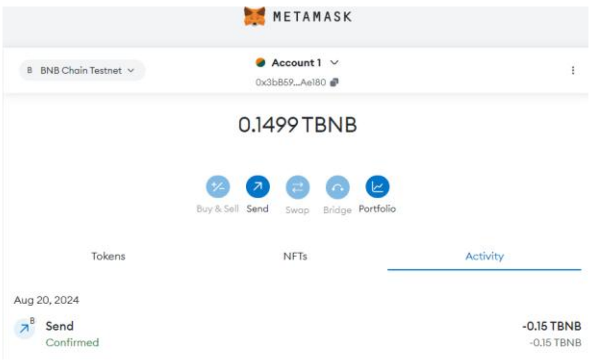
We test by sending 0.15TBNB to another wallet address


Send

Status	View on block explorer
Confirmed	Copy transaction ID
From	To
0x3bB59...A...	0x34944...

Transaction

Nonce	0
Amount	-0.15 TBNB
Gas Limit (Units)	21000
Gas Used (Units)	21000
Base fee (GWEI)	0
Priority fee (GWEI)	3.201
Total gas fee	0.000067 TBNB
Max fee per gas	0.000000003 TBNB
Total	0.15006722 TBNB


METAMASK

BNB Chain Testnet

Account 1

0x3bB59...Ae180

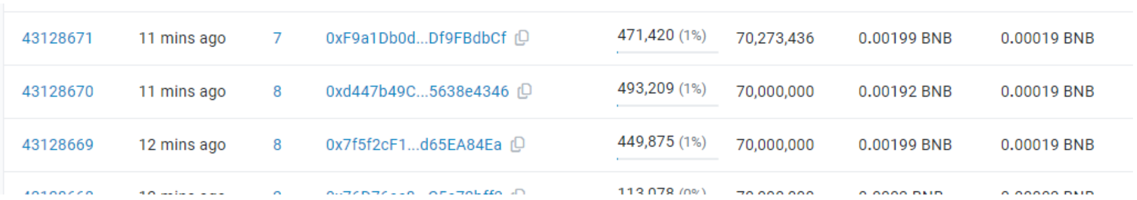
0.1499 TBNB

Aug 20, 2024

Send	-0.15 TBNB
Confirmed	-0.15 TBNB

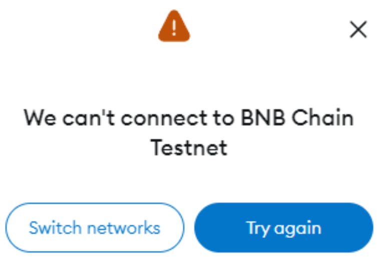
We can see a confirmation of the transaction on the BNB blockchain

https://testnet.bscscan.com/blocks?ps=100&p=3

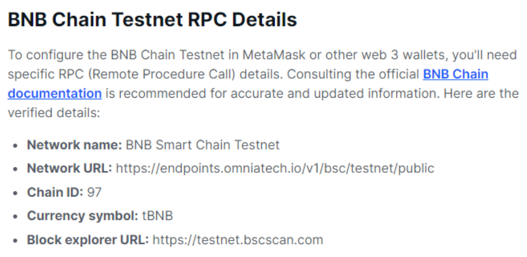


43128671	11 mins ago	7	0xF9a1Db0d...Df9FBdbCf	471,420 (1%)	70,273,436	0.00199 BNB	0.00019 BNB
43128670	11 mins ago	8	0xd447b49C...5638e4346	493,209 (1%)	70,000,000	0.00192 BNB	0.00019 BNB
43128669	12 mins ago	8	0x7f5f2cF1...d65EA84Ea	449,875 (1%)	70,000,000	0.00199 BNB	0.00019 BNB
[illegible]	[illegible]	[illegible]	[illegible]	113,078 (0%)	[illegible]	[illegible]	[illegible]

Attempt to resend BNB testnet how error adding the wallet (created a new Metamask wallet on a different computer – so unable to get TBNB)


We can't connect to BNB Chain Testnet

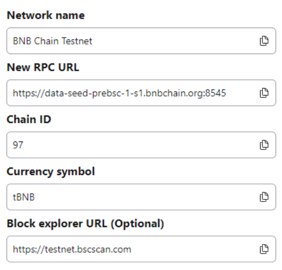
Below one worked


BNB Chain Testnet RPC Details

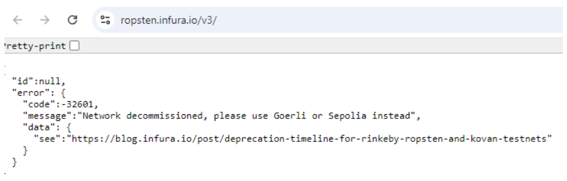
To configure the BNB Chain Testnet in MetaMask or other web 3 wallets, you'll need specific RPC (Remote Procedure Call) details. Consulting the official BNB Chain documentation is recommended for accurate and updated information. Here are the verified details:

- **Network name:** BNB Smart Chain Testnet
- **Network URL:** https://endpoints.omniatech.io/v1/bsc/testnet/public
- **Chain ID:** 97
- **Currency symbol:** tBNB
- **Block explorer URL:** https://testnet.bscscan.com

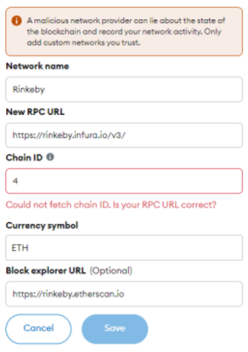
This one below did not work


Network name	BNB Chain Testnet
New RPC URL	https://data-seed-prebsc-1-s1.bnbchain.org:8545
Chain ID	97
Currency symbol	tBNB
Block explorer URL (Optional)	https://testnet.bscscan.com

Ropsten does not work either

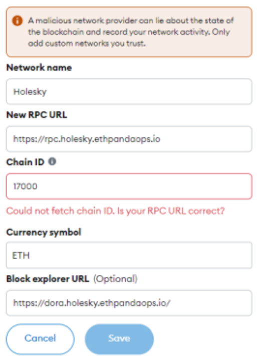

ropsten.infura.io/v3/

```
'retty-print

 "id":null,
 "error": {
   "code":-32601,
   "message":"Network decommissioned, please use Goerli or Sepolia instead",
   "data": {
     "see":"https://blog.infura.io/post/deprecation-timeline-for-rinkeby-ropsten-and-kovan-testnets"
   }
 }
```


A malicious network provider can lie about the state of the blockchain and record your network activity. Only add custom networks you trust.

Network name	Rinkeby
New RPC URL	https://rinkeby.infura.io/v3/
Chain ID	4
Currency symbol	ETH
Block explorer URL (Optional)	https://rinkeby.etherscan.io

Could not fetch chain ID. Is your RPC URL correct?

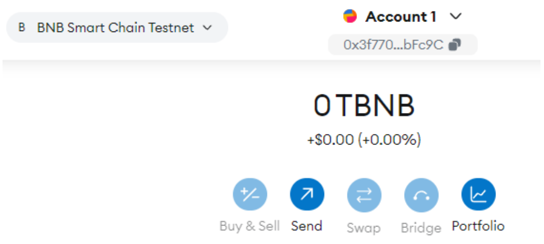

A malicious network provider can lie about the state of the blockchain and record your network activity. Only add custom networks you trust.

Network name	Holesky
New RPC URL	https://rpc.holesky.ethpandaops.io
Chain ID	17000
Currency symbol	ETH
Block explorer URL (Optional)	https://dora.holesky.ethpandaops.io/

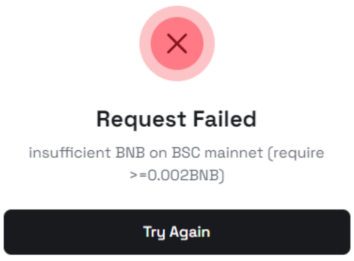
Could not fetch chain ID. Is your RPC URL correct?

- **Sepolia**
- **Goerli** **deprecated*
- **Holešky**
- **Rinkeby** **sunsetted*
- **Ropsten** **sunsetted*

I created the tBNB wallet which correct details and appears to be working in metamask and I will now try and send some tBnB



0 TBNB
+$0.00 (+0.00%)

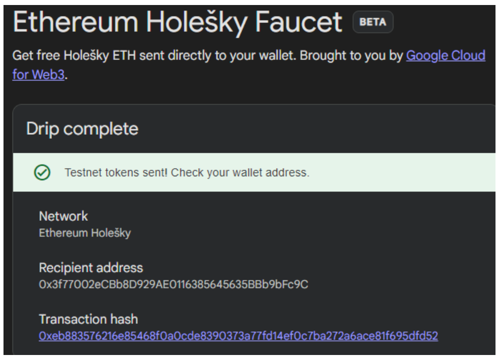


Request Failed

insufficient BNB on BSC mainnet (require >=0.002BNB)

We get an error message from TNB faucet (which did not show before) when I previously sent some in first half of lab.

I create a Holesky Wallet in metamask *which adds successfully and then try using Google Cloud faucet to send some test ethers (Holesky) and this appears to have sent to my wallet address



Ethereum Holešky Faucet BETA

Get free Holešky ETH sent directly to your wallet. Brought to you by Google Cloud for Web3.

Drip complete

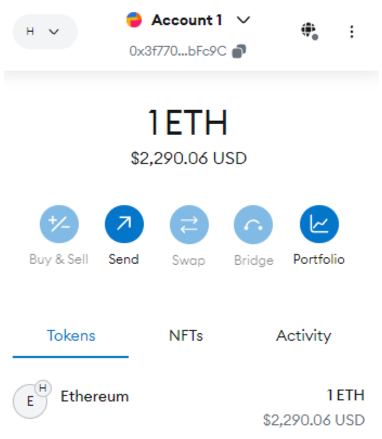
Testnet tokens sent! Check your wallet address.

Network
Ethereum Holešky

Recipient address
0x3f77002eCBb8D929AE0116385645635BBb9bFc9C

Transaction hash
0xeb883576216e85468f0a0cde8390373a77fd14ef0c7ba272a6ace81f695dfd52

I check my Metamask wallet and Woohoo- after several attempts we now finally have some testnet ETH

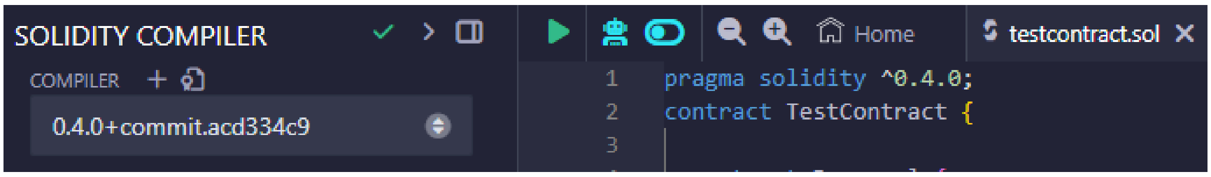


1 ETH
$2,290.06 USD

Ethereum 1 ETH
$2,290.06 USD

So I return to Step 3: Load a Hello World

(Making sure compiler matches pragma solidity version (0.4.0)


SOLIDITY COMPILER

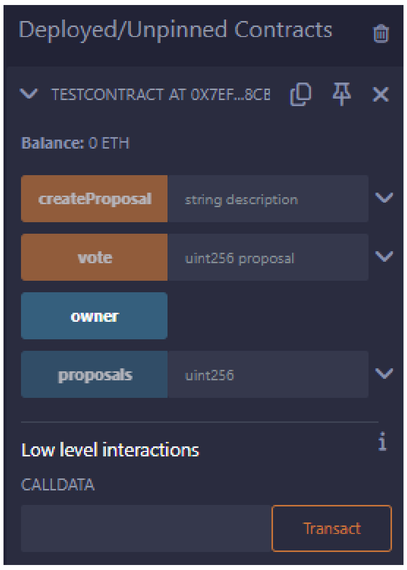
COMPILER

0.4.0+commit.acd334c9

testcontract.sol

```
pragma solidity ^0.4.0;
contract TestContract {
```

We deploy the contract in the Remix VM


Deployed/Unpinned Contracts

TESTCONTRACT AT 0X7EF...8CE

Balance: 0 ETH

createProposal — string description

vote — uint256 proposal

owner

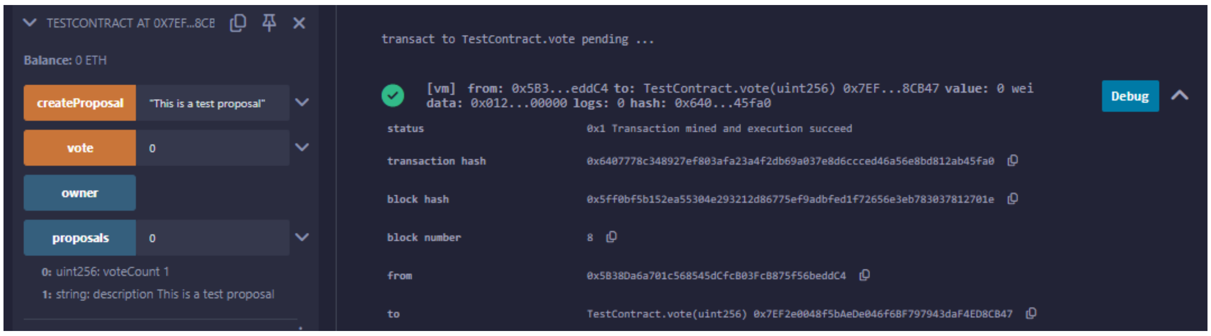
proposals — uint256

Low level interactions

CALLDATA

Transact

Interacting with the test contract and filling out fields


TESTCONTRACT AT 0X7EF...8CE

Balance: 0 ETH

createProposal — "This is a test proposal"

vote — 0

owner

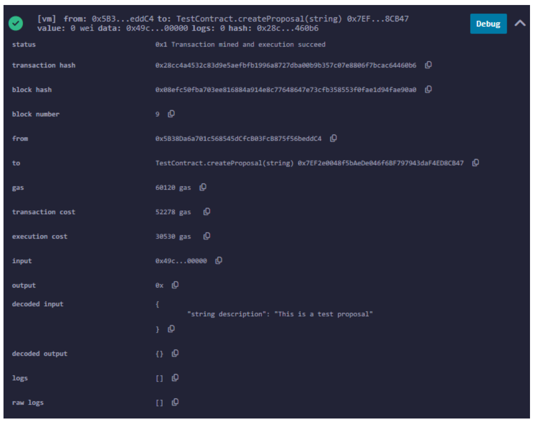
proposals — 0

0: uint256: voteCount 1

1: string: description This is a test proposal

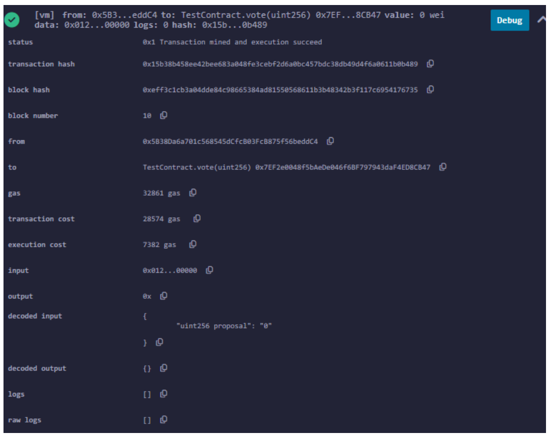
```
transact to TestContract.vote pending ...

[vm]  from: 0x5B3...eddC4 to: TestContract.vote(uint256) 0x7EF...8CB47 value: 0 wei
data: 0x012...00000 logs: 0 hash: 0x640...45fa0
status                 0x1 Transaction mined and execution succeed
transaction hash       0x6407778c348927ef803afa23a4f2db69a037e8d6ccced46a56e8bd812ab45fa0
block hash             0x5ff0bf5b152ea55304e293212d86775ef9adbfed1f72656e3eb783037812701e
block number           8
from                   0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                     TestContract.vote(uint256) 0x7EF2e0048f5bAeDe046f6BF797943daF4ED8CB47
```

Response when clicking createProposal



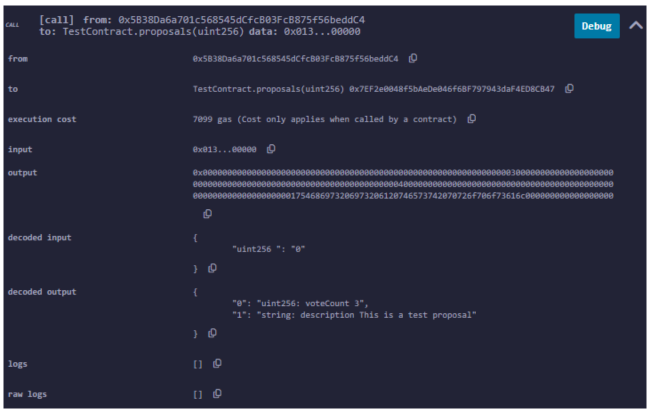
```
[vm]  from: 0x5B3...eddC4 to: TestContract.createProposal(string) 0x7EF...8CB47
value: 0 wei data: 0x49c...00000 logs: 0 hash: 0x28c...460b6
status                0x1 Transaction mined and execution succeed
transaction hash      0x28cc4a4532c83d9e5aefbfb1996a8727dba00b9b357c07e8806f7bcac64460b6
block hash            0x08efc50fba703ee816884a914e8c77648647e73cfb358553f0fae1d94fae90a0
block number          9
from                  0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                    TestContract.createProposal(string) 0x7EF2e0048f5bAeDe046f6BF797943daF4ED8CB47
gas                   60120 gas
transaction cost      52278 gas
execution cost        30530 gas
input                 0x49c...00000
output                0x
decoded input         {
                              "string description": "This is a test proposal"
                      }
decoded output        {}
logs                  []
raw logs              []
```

Response when clicking vote



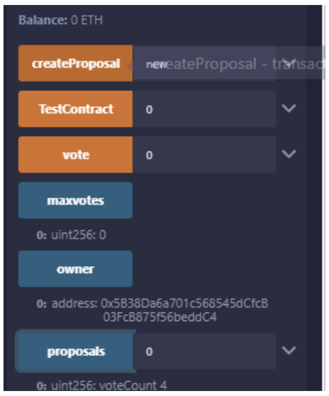
```
[vm]  from: 0x5B3...eddC4 to: TestContract.vote(uint256) 0x7EF...8CB47 value: 0 wei
data: 0x012...00000 logs: 0 hash: 0x15b...0b489
status                0x1 Transaction mined and execution succeed
transaction hash      0x15b38b458ee42bee683a048fe3cebf2d6a0bc457bdc38db49d4f6a0611b0b489
block hash            0xeff3c1cb3a04dde84c98665384ad81550568611b3b48342b3f117c6954176735
block number          10
from                  0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                    TestContract.vote(uint256) 0x7EF2e0048f5bAeDe046f6BF797943daF4ED8CB47
gas                   32861 gas
transaction cost      28574 gas
execution cost        7382 gas
input                 0x012...00000
output                0x
decoded input         {
                              "uint256 proposal": "0"
                      }
decoded output        {}
logs                  []
raw logs              []
```

Response when clicking proposals

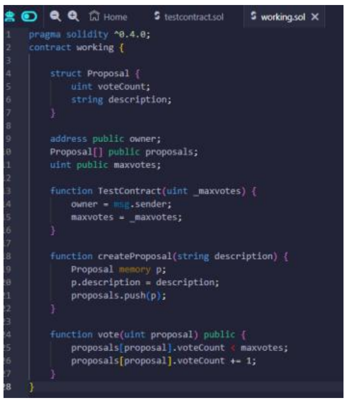


```
CALL   [call]  from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
       to: TestContract.proposals(uint256) data: 0x013...00000
from                  0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                    TestContract.proposals(uint256) 0x7EF2e0048f5bAeDe046f6BF797943daF4ED8CB47
execution cost        7099 gas (Cost only applies when called by a contract)
input                 0x013...00000
output                0x000000000000000000000000000000000000000000000000000000000000000300000000000000000000
                      000000000000000000000000000000000000000000400000000000000000000000000000000000000000000
                      0000000000000000000017546869732069732061207465737420707270706f73616c000000000000000000
decoded input         {
                              "uint256 ": "0"
                      }
decoded output        {
                              "0": "uint256: voteCount 3",
                              "1": "string: description This is a test proposal"
                      }
logs                  []
raw logs              []
```

An exercise to add max voting (the solidity is 0.4.0 so very outdated)

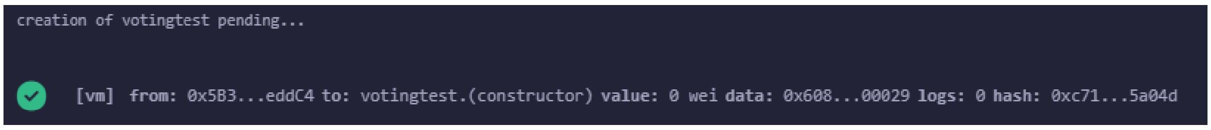


```
Balance: 0 ETH
createProposal   neweateProposal - transac
TestContract     0
vote             0
maxvotes
0: uint256: 0
owner
0: address: 0x5B38Da6a701c568545dCfcB
   03FcB875f56beddC4
proposals        0
0: uint256: voteCount 4
```

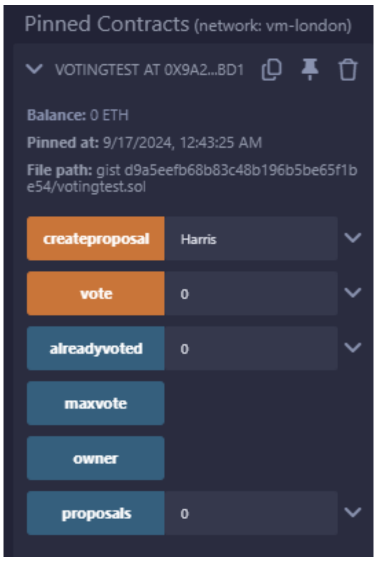


```
pragma solidity ^0.4.0;
contract working {

    struct Proposal {
        uint voteCount;
        string description;
    }

    address public owner;
    Proposal[] public proposals;
    uint public maxvotes;

    function TestContract(uint _maxvotes) {
        owner = msg.sender;
        maxvotes = _maxvotes;
    }

    function createProposal(string description) {
        Proposal memory p;
        p.description = description;
        proposals.push(p);
    }

    function vote(uint proposal) public {
        proposals[proposal].voteCount < maxvotes;
        proposals[proposal].voteCount += 1;
    }
}
```

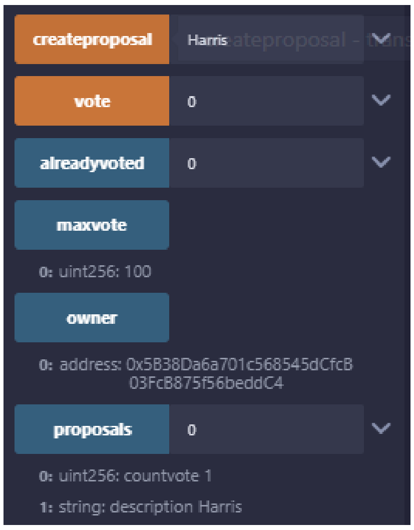
We have created a new smart contract called votingtest.sol under a new compiler 0.4.24



```
creation of votingtest pending...

[vm]  from: 0x5B3...eddC4 to: votingtest.(constructor) value: 0 wei data: 0x608...00029 logs: 0 hash: 0xc71...5a04d
```

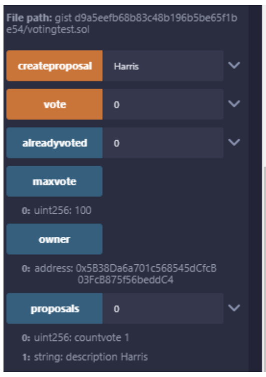


```
Pinned Contracts (network: vm-london)
VOTINGTEST AT 0X9A2...BD1

Balance: 0 ETH
Pinned at: 9/17/2024, 12:43:25 AM
File path: gist d9a5eefb68b83c48b196b5be65f1b
e54/votingtest.sol

createproposal   Harris
vote             0
alreadyvoted     0
maxvote
owner
proposals        0
```

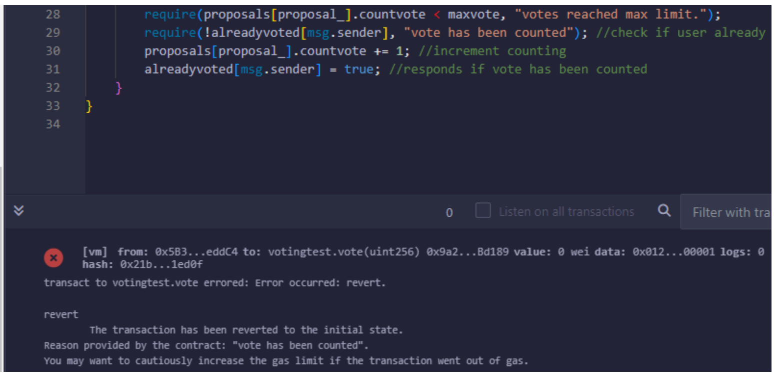


```
createproposal   Harris
vote             0
alreadyvoted     0
maxvote
0: uint256: 100
owner
0: address: 0x5B38Da6a701c568545dCfcB
   03FcB875f56beddC4
proposals        0
0: uint256: countvote 1
1: string: description Harris
```

We receive an error message saying "vote has been counted" in the console when trying to vote again as per our vote function. This smart contract has recognized our address and will not record another vote.



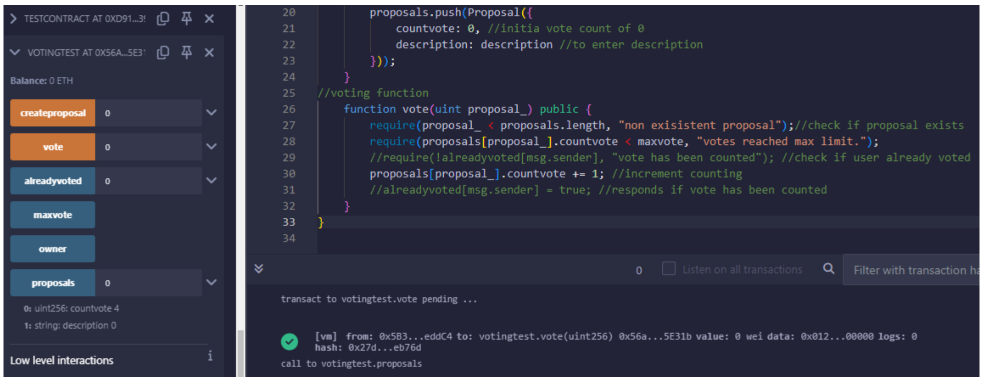
```
File path: gist d9a5eefb68b83c48b196b5be65f1b
e54/votingtest.sol
createproposal   Harris
vote             0
alreadyvoted     0
maxvote
0: uint256: 100
owner
0: address: 0x5B38Da6a701c568545dCfcB
   03FcB875f56beddC4
proposals        0
0: uint256: countvote 1
1: string: description Harris
```



```
            require(proposals[proposal_].countvote < maxvote, "votes reached max limit.");
            require(!alreadyvoted[msg.sender], "vote has been counted"); //check if user already
            proposals[proposal_].countvote += 1; //increment counting
            alreadyvoted[msg.sender] = true; //responds if vote has been counted
        }
    }
```

```
[vm]  from: 0x5B3...eddC4 to: votingtest.vote(uint256) 0x9a2...Bd189 value: 0 wei data: 0x012...00001 logs: 0
hash: 0x21b...1ed0f
transact to votingtest.vote errored: Error occurred: revert.

revert
        The transaction has been reverted to the initial state.
Reason provided by the contract: "vote has been counted".
You may want to cautiously increase the gas limit if the transaction went out of gas.
```

When testing the vote count after commenting the two lines to check votes, the vote can be pressed multiple times and accounted for

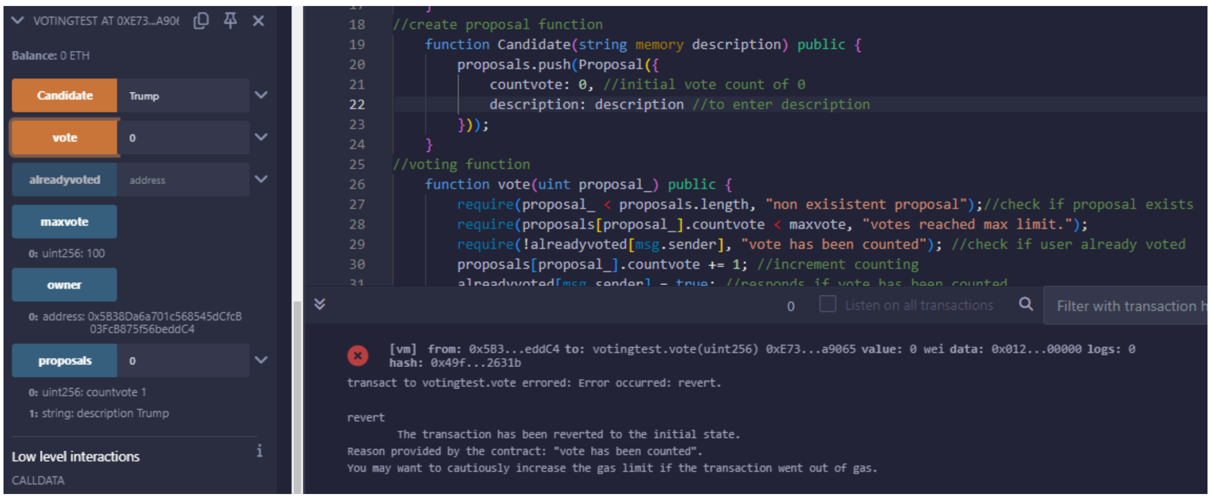


TESTCONTRACT AT 0XD91...3!
VOTINGTEST AT 0X56A...5E31
Balance: 0 ETH
createproposal 0
vote 0
alreadyvoted 0
maxvote
owner
proposals 0
0: uint256: countvote 4
1: string: description 0
Low level interactions

```
        proposals.push(Proposal({
            countvote: 0, //initia vote count of 0
            description: description //to enter description
        }));
    }
//voting function
    function vote(uint proposal_) public {
        require(proposal_ < proposals.length, "non exisistent proposal");//check if proposal exists
        require(proposals[proposal_].countvote < maxvote, "votes reached max limit.");
        //require(!alreadyvoted[msg.sender], "vote has been counted"); //check if user already voted
        proposals[proposal_].countvote += 1; //increment counting
        //alreadyvoted[msg.sender] = true; //responds if vote has been counted
    }
}
```

```
transact to votingtest.vote pending ...

[vm] from: 0x5B3...eddC4 to: votingtest.vote(uint256) 0x56a...5E31b value: 0 wei data: 0x012...00000 logs: 0
hash: 0x27d...eb76d
call to votingtest.proposals
```

Few changes to make it more elective type

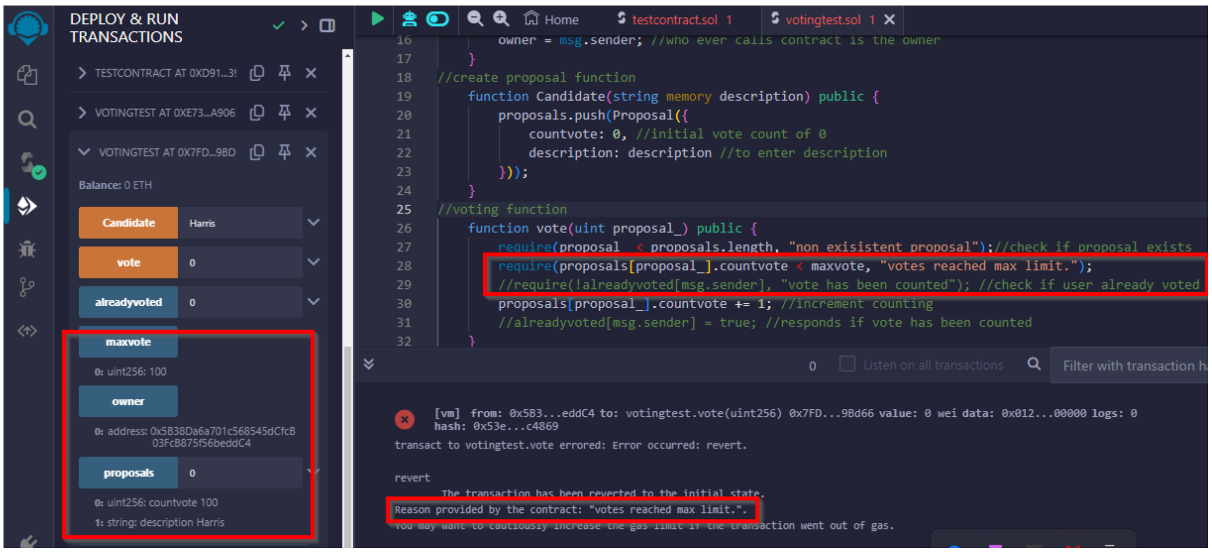


VOTINGTEST AT 0XE73...A90(
Balance: 0 ETH
Candidate Trump
vote 0
alreadyvoted address
maxvote
0: uint256: 100
owner
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
proposals 0
0: uint256: countvote 1
1: string: description Trump
Low level interactions
CALLDATA

```
//create proposal function
    function Candidate(string memory description) public {
        proposals.push(Proposal({
            countvote: 0, //initial vote count of 0
            description: description //to enter description
        }));
    }
//voting function
    function vote(uint proposal_) public {
        require(proposal_ < proposals.length, "non exisistent proposal");//check if proposal exists
        require(proposals[proposal_].countvote < maxvote, "votes reached max limit.");
        require(!alreadyvoted[msg.sender], "vote has been counted"); //check if user already voted
        proposals[proposal_].countvote += 1; //increment counting
        alreadyvoted[msg.sender] = true; //responds if vote has been counted
```

```
[vm] from: 0x5B3...eddC4 to: votingtest.vote(uint256) 0xE73...a9065 value: 0 wei data: 0x012...00000 logs: 0
hash: 0x49f...2631b
transact to votingtest.vote errored: Error occurred: revert.

revert
        The transaction has been reverted to the initial state.
Reason provided by the contract: "vote has been counted".
You may want to cautiously increase the gas limit if the transaction went out of gas.
```

We test the 100 votes max limit while commenting out the one vote per user and we get an error message when trying to vote more than 100 times “votes reached max limit”

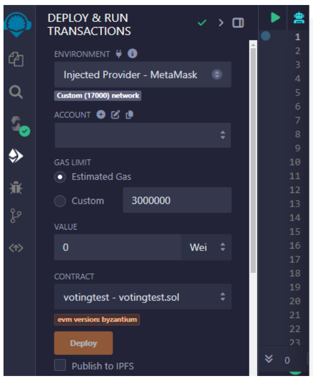


DEPLOY & RUN TRANSACTIONS
TESTCONTRACT AT 0XD91...3!
VOTINGTEST AT 0XE73...A906
VOTINGTEST AT 0X7FD...9BD
Balance: 0 ETH
Candidate Harris
vote 0
alreadyvoted 0
maxvote
0: uint256: 100
owner
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
proposals 0
0: uint256: countvote 100
1: string: description Harris

```
        owner = msg.sender; //who ever calls contract is the owner
    }
//create proposal function
    function Candidate(string memory description) public {
        proposals.push(Proposal({
            countvote: 0, //initial vote count of 0
            description: description //to enter description
        }));
    }
//voting function
    function vote(uint proposal_) public {
        require(proposal_ < proposals.length, "non exisistent proposal");//check if proposal exists
        require(proposals[proposal_].countvote < maxvote, "votes reached max limit.");
        //require(!alreadyvoted[msg.sender], "vote has been counted"); //check if user already voted
        proposals[proposal_].countvote += 1; //increment counting
        //alreadyvoted[msg.sender] = true; //responds if vote has been counted
    }
```

```
[vm] from: 0x5B3...eddC4 to: votingtest.vote(uint256) 0x7FD...9Bd66 value: 0 wei data: 0x012...00000 logs: 0
hash: 0x53e...c4869
transact to votingtest.vote errored: Error occurred: revert.

revert
        The transaction has been reverted to the initial state.
Reason provided by the contract: "votes reached max limit.".
You may want to cautiously increase the gas limit if the transaction went out of gas.
```

Step 4: Publish the votingtest contract to the Testnet


DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Injected Provider - MetaMask
Custom (17000) network

ACCOUNT

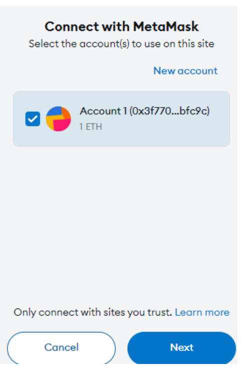
GAS LIMIT
● Estimated Gas
○ Custom 3000000

VALUE: 0 Wei

CONTRACT: votingtest - votingtest.sol
evm version: byzantium

Deploy

☐ Publish to IPFS

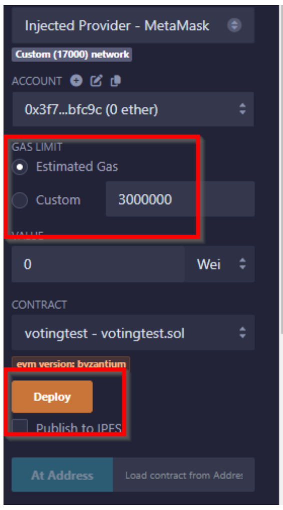

Connect with MetaMask
Select the account(s) to use on this site

New account

☑ Account 1 (0x3f770...bfc9c)
1 ETH

Only connect with sites you trust. Learn more

Cancel | Next


Injected Provider - MetaMask
Custom (17000) network

ACCOUNT: 0x3f7...bfc9c (0 ether)

GAS LIMIT
● Estimated Gas
○ Custom 3000000

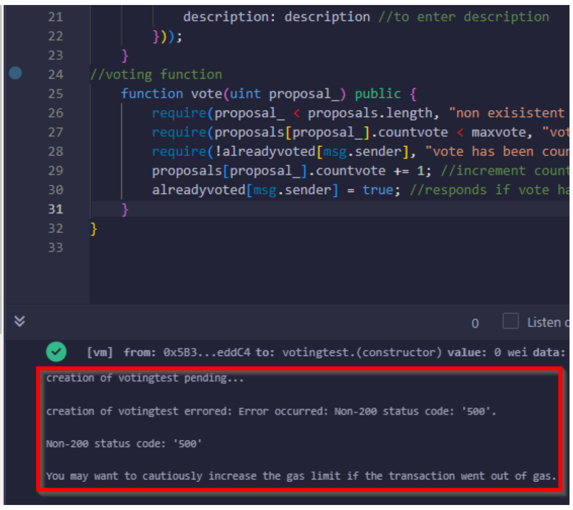
VALUE: 0 Wei

CONTRACT: votingtest - votingtest.sol
evm version: byzantium

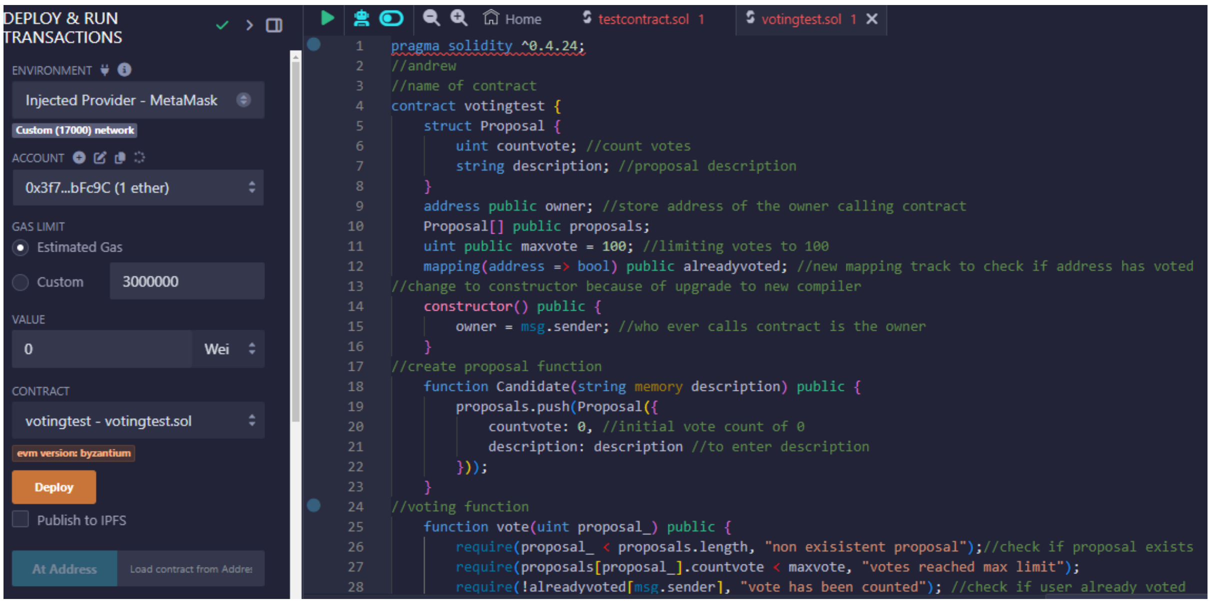
Deploy

☐ Publish to IPFS

At Address | Load contract from Address


```
                description: description //to enter description
            }));
        }
//voting function
    function vote(uint proposal_) public {
        require(proposal_ < proposals.length, "non exisistent
        require(proposals[proposal_].countvote < maxvote, "vot
        require(!alreadyvoted[msg.sender], "vote has been cour
        proposals[proposal_].countvote += 1; //increment count
        alreadyvoted[msg.sender] = true; //responds if vote ha
    }
}
```

```
[vm] from: 0x5B3...eddC4 to: votingtest.(constructor) value: 0 wei data:
creation of votingtest pending...

creation of votingtest errored: Error occurred: Non-200 status code: '500'.

Non-200 status code: '500'

You may want to cautiously increase the gas limit if the transaction went out of gas.
```


DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Injected Provider - MetaMask
Custom (17000) network

ACCOUNT: 0x3f7...bFc9C (1 ether)

GAS LIMIT
● Estimated Gas
○ Custom 3000000

VALUE: 0 Wei

CONTRACT: votingtest - votingtest.sol
evm version: byzantium

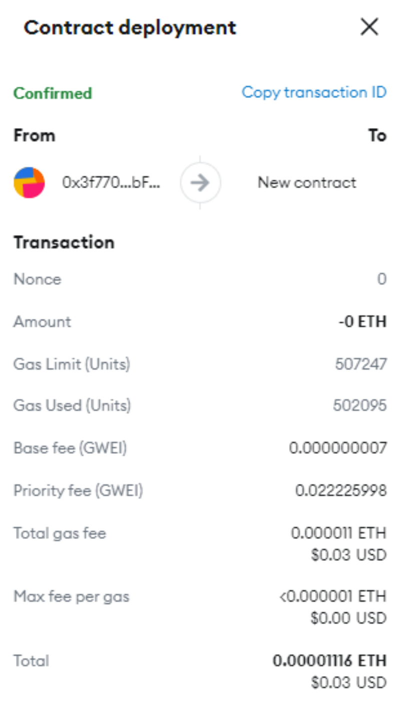
Deploy

☐ Publish to IPFS

At Address | Load contract from Address

Home | testcontract.sol 1 | votingtest.sol 1

```
pragma solidity ^0.4.24;
//andrew
//name of contract
contract votingtest {
    struct Proposal {
        uint countvote; //count votes
        string description; //proposal description
    }
    address public owner; //store address of the owner calling contract
    Proposal[] public proposals;
    uint public maxvote = 100; //limiting votes to 100
    mapping(address => bool) public alreadyvoted; //new mapping track to check if address has voted
//change to constructor because of upgrade to new compiler
    constructor() public {
        owner = msg.sender; //who ever calls contract is the owner
    }
//create proposal function
    function Candidate(string memory description) public {
        proposals.push(Proposal({
            countvote: 0, //initial vote count of 0
            description: description //to enter description
        }));
    }
//voting function
    function vote(uint proposal_) public {
        require(proposal_ < proposals.length, "non exisistent proposal");//check if proposal exists
        require(proposals[proposal_].countvote < maxvote, "votes reached max limit");
        require(!alreadyvoted[msg.sender], "vote has been counted"); //check if user already voted
```


Contract deployment

Confirmed | Copy transaction ID

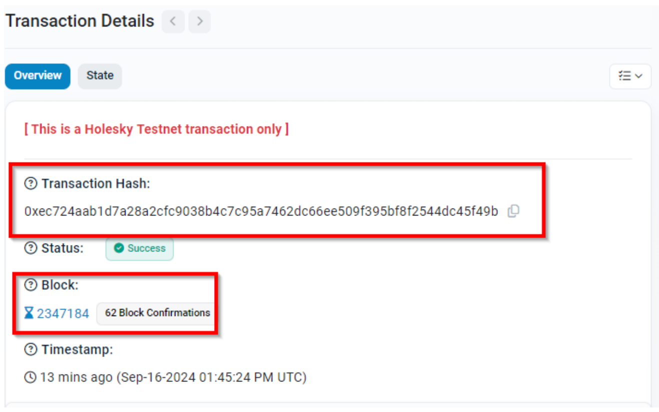
From	To
0x3f770...bF...	New contract

Transaction

Nonce	0
Amount	-0 ETH
Gas Limit (Units)	507247
Gas Used (Units)	502095
Base fee (GWEI)	0.000000007
Priority fee (GWEI)	0.022225998
Total gas fee	0.000011 ETH $0.03 USD
Max fee per gas	<0.000001 ETH $0.00 USD
Total	0.00001116 ETH $0.03 USD

We can see the confirmation of the transaction on the Ethernet scan page

https://holesky.etherscan.io/address/0x90ed6e28cf58ebdec0f8b98d2954b2ccdc576896



Transaction Details

Overview State

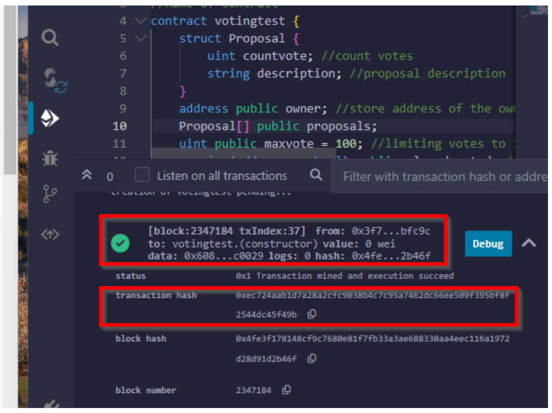
[This is a Holesky Testnet transaction only]

Transaction Hash:
0xec724aab1d7a28a2cfc9038b4c7c95a7462dc66ee509f395bf8f2544dc45f49b

Status: Success

Block:
2347184 62 Block Confirmations

Timestamp:
13 mins ago (Sep-16-2024 01:45:24 PM UTC)

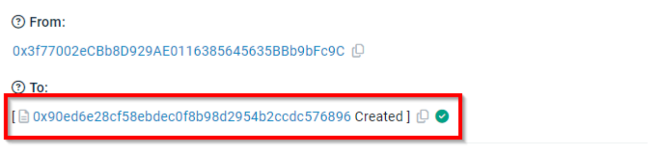


```
contract votingtest {
    struct Proposal {
        uint countvote; //count votes
        string description; //proposal description
    }
    address public owner; //store address of the ow
    Proposal[] public proposals;
    uint public maxvote = 100; //limiting votes to
```

0 Listen on all transactions Filter with transaction hash or addre

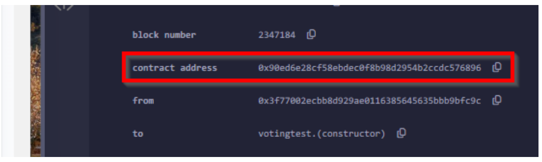
```
[block:2347184 txIndex:37] from: 0x3f7...bfc9c
to: votingtest.(constructor) value: 0 wei
data: 0x608...c0029 logs: 0 hash: 0x4fe...2b46f
status            0x1 Transaction mined and execution succeed
transaction hash  0xec724aab1d7a28a2cfc9038b4c7c95a7462dc66ee509f395bf8f
                  2544dc45f49b
block hash        0x4fe3f178148cf9c7680e81f7fb33a3ae688330aa4eec116a1972
                  d28d91d2b46f
block number      2347184
```

Debug

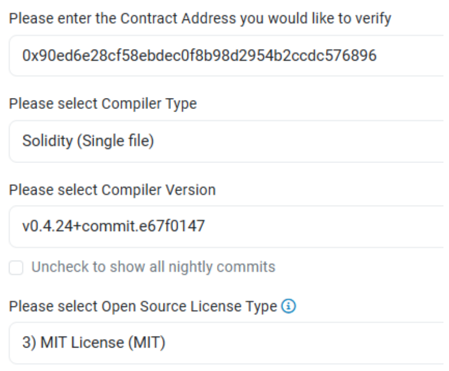


From:
0x3f77002eCBb8D929AE0116385645635BBb9bFc9C

To:
[0x90ed6e28cf58ebdec0f8b98d2954b2ccdc576896 Created]



```
block number      2347184
contract address  0x90ed6e28cf58ebdec0f8b98d2954b2ccdc576896
from              0x3f77002ecbb8d929ae0116385645635bbb9bfc9c
to                votingtest.(constructor)
```



Please enter the Contract Address you would like to verify
0x90ed6e28cf58ebdec0f8b98d2954b2ccdc576896

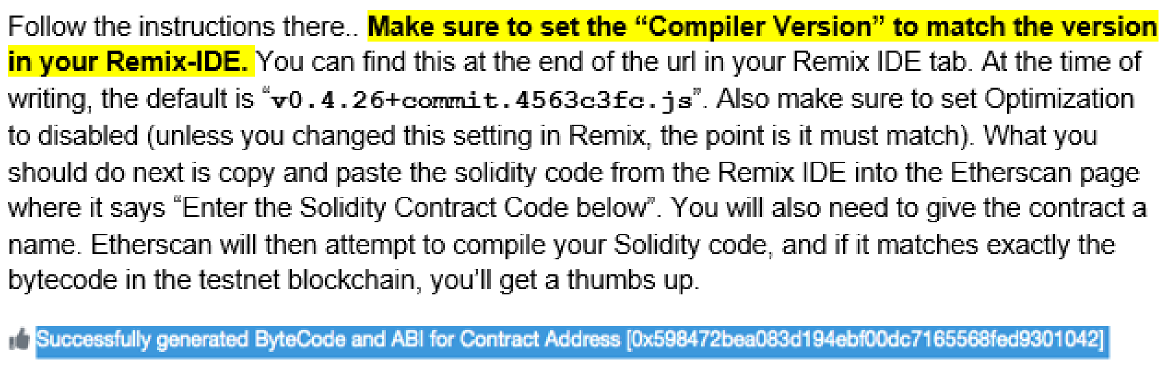
Please select Compiler Type
Solidity (Single file)

Please select Compiler Version
v0.4.24+commit.e67f0147

Uncheck to show all nightly commits

Please select Open Source License Type
3) MIT License (MIT)

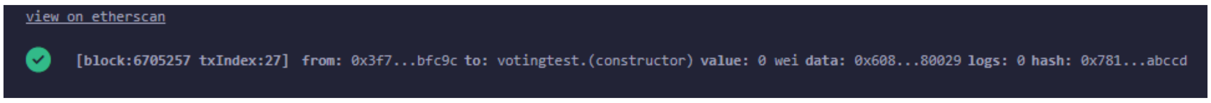
This did not work when verifying and publishing



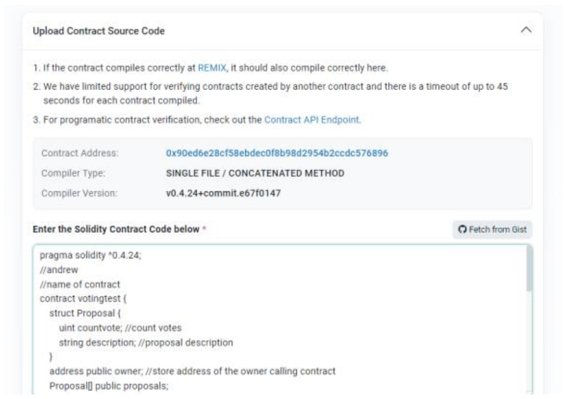
Follow the instructions there.. **Make sure to set the "Compiler Version" to match the version in your Remix-IDE.** You can find this at the end of the url in your Remix IDE tab. At the time of writing, the default is "`v0.4.26+commit.4563c3fc.js`". Also make sure to set Optimization to disabled (unless you changed this setting in Remix, the point is it must match). What you should do next is copy and paste the solidity code from the Remix IDE into the Etherscan page where it says "Enter the Solidity Contract Code below". You will also need to give the contract a name. Etherscan will then attempt to compile your Solidity code, and if it matches exactly the bytecode in the testnet blockchain, you'll get a thumbs up.

Successfully generated ByteCode and ABI for Contract Address [0x598472bea083d194ebf00dc7165568fed9301042]

We try this again



```
view on etherscan

[block:6705257 txIndex:27] from: 0x3f7...bfc9c to: votingtest.(constructor) value: 0 wei data: 0x608...80029 logs: 0 hash: 0x781...abccd
```



Upload Contract Source Code

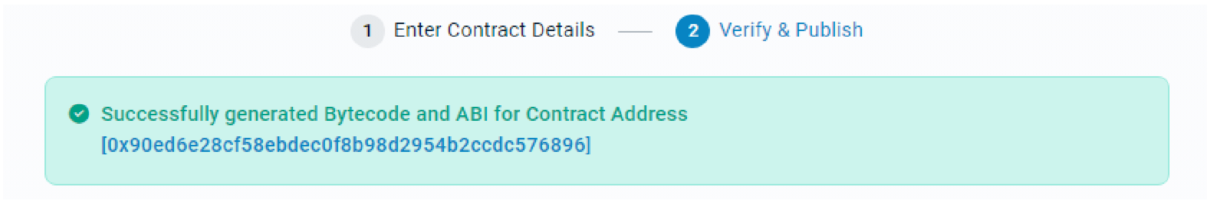
1. If the contract compiles correctly at REMIX, it should also compile correctly here.
2. We have limited support for verifying contracts created by another contract and there is a timeout of up to 45 seconds for each contract compiled.
3. For programatic contract verification, check out the Contract API Endpoint.

Contract Address:	0x90ed6e28cf58ebdec0f8b98d2954b2ccdc576896
Compiler Type:	SINGLE FILE / CONCATENATED METHOD
Compiler Version:	v0.4.24+commit.e67f0147

Enter the Solidity Contract Code below *

```
pragma solidity ^0.4.24;
//andrew
//name of contract
contract votingtest {
   struct Proposal {
      uint countvote; //count votes
      string description; //proposal description
   }
   address public owner; //store address of the owner calling contract
   Proposal[] public proposals;
```

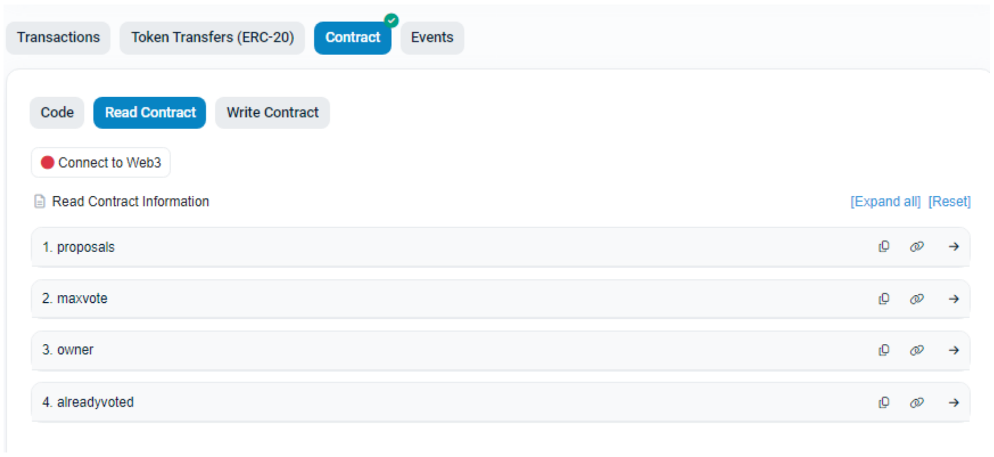
Woo-hoo we are successful in verifying the “votingtest” smart contract



Successfully generated Bytecode and ABI for Contract Address [0x90ed6e28cf58ebdec0f8b98d2954b2ccdc576896]

https://sepolia.etherscan.io/address/0x90ed6e28cf58ebdec0f8b98d2954b2ccdc576896#code

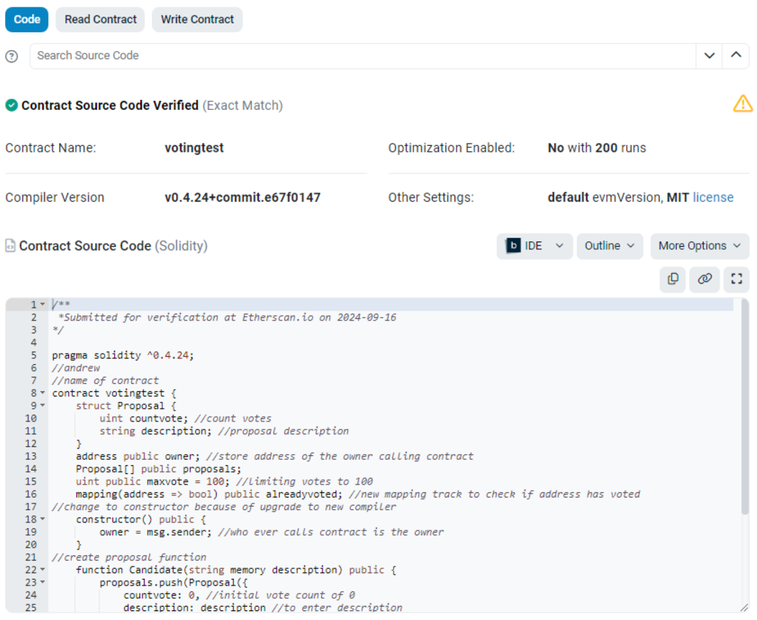
This is the read contract tab, and can see the public fields in the “votingtest” smart contract



1. proposals
2. maxvote
3. owner
4. alreadyvoted

Step 6: Share contract with someone else or use someone else's contract

Sharing via verified source code



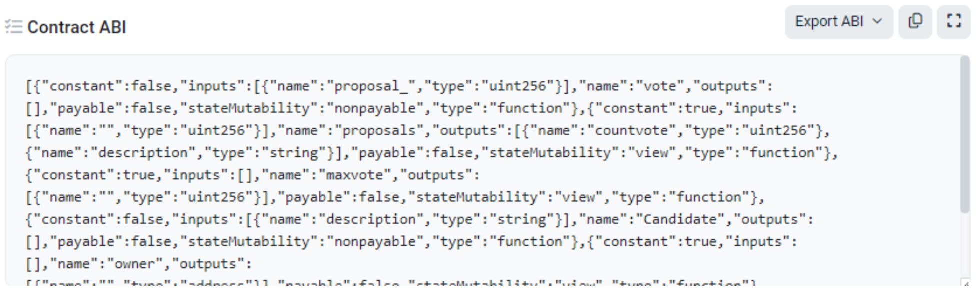
Contract Source Code Verified (Exact Match)

Contract Name:	votingtest	Optimization Enabled:	No with 200 runs
Compiler Version	v0.4.24+commit.e67f0147	Other Settings:	default evmVersion, MIT license

Contract Source Code (Solidity)

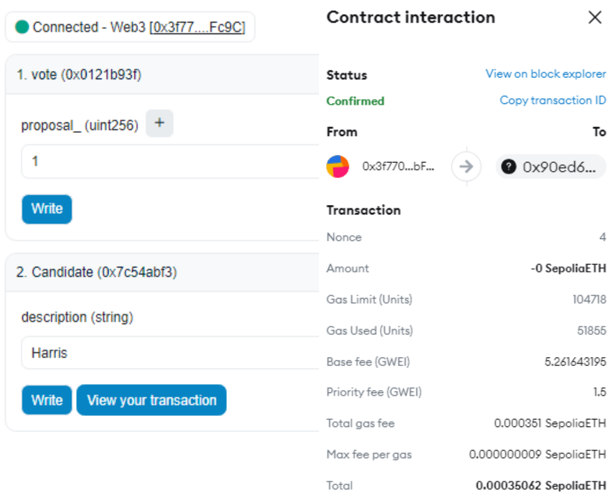
```
/**
 *Submitted for verification at Etherscan.io on 2024-09-16
*/

pragma solidity ^0.4.24;
//andrew
//name of contract
contract votingtest {
    struct Proposal {
        uint countvote; //count votes
        string description; //proposal description
    }
    address public owner; //store address of the owner calling contract
    Proposal[] public proposals;
    uint public maxvote = 100; //limiting votes to 100
    mapping(address => bool) public alreadyvoted; //new mapping track to check if address has voted
//change to constructor because of upgrade to new compiler
    constructor() public {
        owner = msg.sender; //who ever calls contract is the owner
    }
//create proposal function
    function Candidate(string memory description) public {
        proposals.push(Proposal({
            countvote: 0, //initial vote count of 0
            description: description //to enter description
```

Sharing via ABI


Contract ABI

```
[{"constant":false,"inputs":[{"name":"proposal_","type":"uint256"}],"name":"vote","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},{"constant":true,"inputs":
[{"name":"","type":"uint256"}],"name":"proposals","outputs":[{"name":"countvote","type":"uint256"},
{"name":"description","type":"string"}],"payable":false,"stateMutability":"view","type":"function"},
{"constant":true,"inputs":[],"name":"maxvote","outputs":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability":"view","type":"function"},
{"constant":false,"inputs":[{"name":"description","type":"string"}],"name":"Candidate","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},{"constant":true,"inputs":
[],"name":"owner","outputs":
[{"name":"","type":"address"}],"payable":false,"stateMutability":"view","type":"function"}
```

We have tested this and below we can see the contract interaction and transactions


Connected - Web3 [0x3f77....Fc9C]

1. vote (0x0121b93f)

proposal_ (uint256)

1

2. Candidate (0x7c54abf3)

description (string)

Harris

Contract interaction

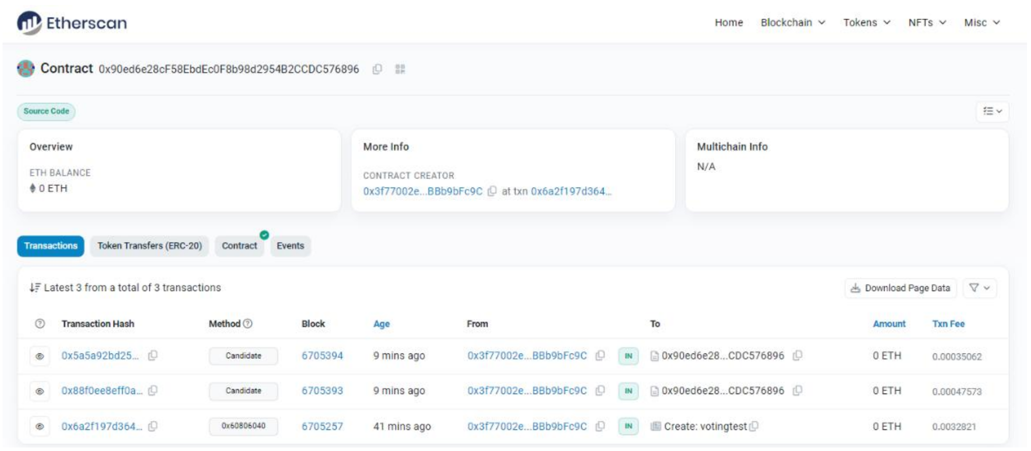
Status: Confirmed

From: 0x3f770...bF...
To: 0x90ed6...

Transaction

Nonce	4
Amount	-0 SepoliaETH
Gas Limit (Units)	104718
Gas Used (Units)	51855
Base fee (GWEI)	5.261643195
Priority fee (GWEI)	1.5
Total gas fee	0.000351 SepoliaETH
Max fee per gas	0.000000009 SepoliaETH
Total	0.00035062 SepoliaETH

Below we can see the transaction made when writing to the “candidate” function of the smart contract. The above fees were charged.



Contract 0x90ed6e28cF58EbdEc0F8b98d2954B2CCDC576896

Overview

ETH BALANCE
0 ETH

More Info

CONTRACT CREATOR
0x3f77002e...BBb9bFc9C at txn 0x6a2f197d364...

Multichain Info
N/A

Latest 3 from a total of 3 transactions

Transaction Hash	Method	Block	Age	From		To	Amount	Txn Fee
0x5a5a92bd25...	Candidate	6705394	9 mins ago	0x3f77002e...BBb9bFc9C	IN	0x90ed6e28...CDC576896	0 ETH	0.00035062
0x88f0ee8eff0a...	Candidate	6705393	9 mins ago	0x3f77002e...BBb9bFc9C	IN	0x90ed6e28...CDC576896	0 ETH	0.00047573
0x6a2f197d364...	0x60806040	6705257	41 mins ago	0x3f77002e...BBb9bFc9C	IN	Create: votingtest	0 ETH	0.0032821

```
pragma solidity ^0.4.24;
//andrew
//name of contract
contract votingtest {
  struct Proposal {
    uint countvote; //count votes
    string description; //proposal description
  }
  address public owner; //store address of the owner calling contract
  Proposal[] public proposals;
  uint public maxvote = 100; //limiting votes to 100
  mapping(address => bool) public alreadyvoted; //new mapping track to check if address
has voted

//change to constructor because of upgrade to new compiler
  constructor() public {
    owner = msg.sender; //who ever calls contract is the owner
  }
//create proposal function
  function Candidate(string memory description) public {
    proposals.push(Proposal({
      countvote: 0, //initial vote count of 0
      description: description //to enter description
    }));
  }
//voting function
  function vote(uint proposal_) public {
    require(proposal_ < proposals.length, "non exisistent proposal");//check if proposal
exists
    require(proposals[proposal_].countvote < maxvote, "votes reached max limit.");
    require(!alreadyvoted[msg.sender], "vote has been counted"); //check if user already
voted
    proposals[proposal_].countvote += 1; //increment counting
    alreadyvoted[msg.sender] = true; //responds if vote has been counted
  }
}
```