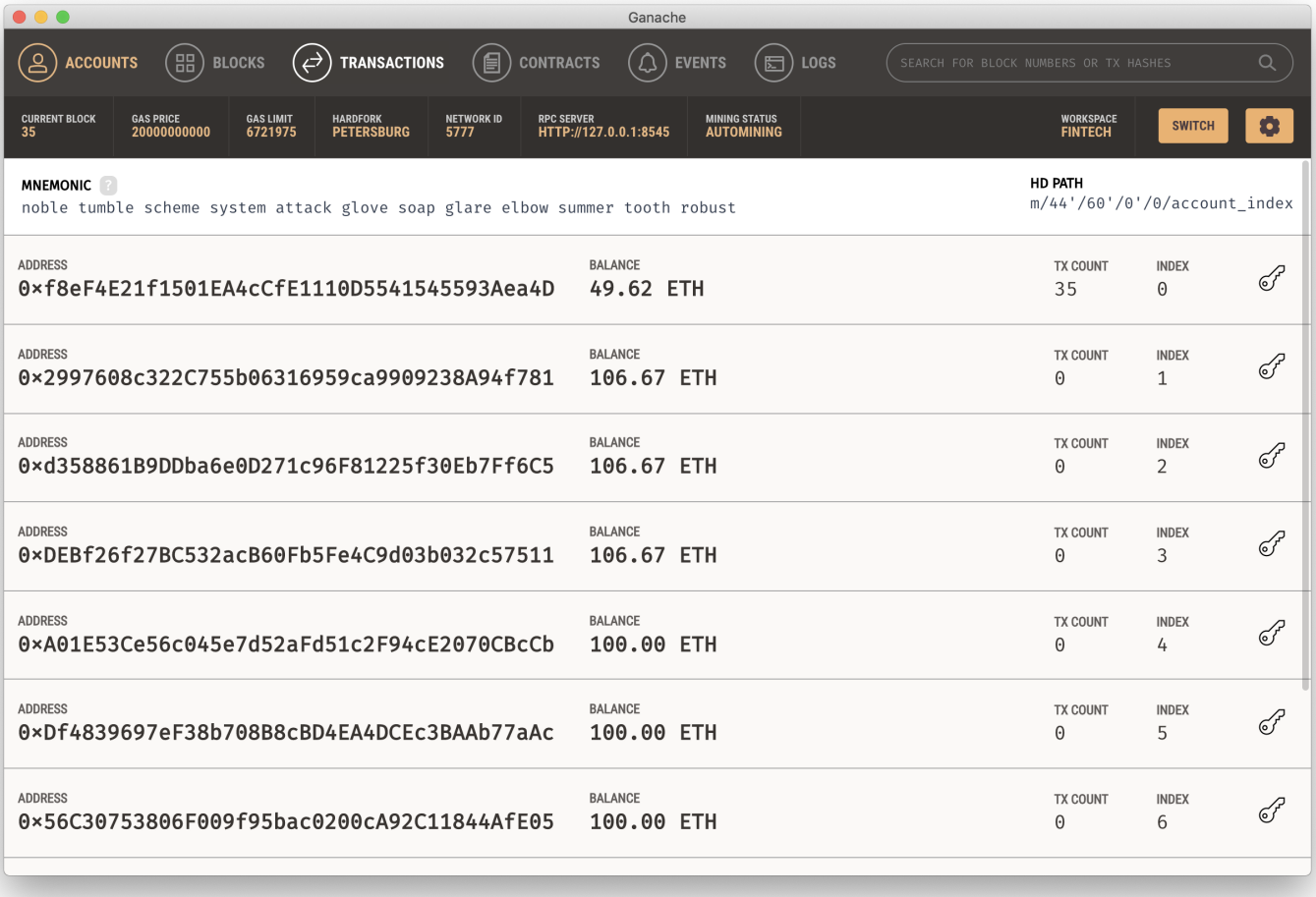
Profit Splitter Smart Contracts

In this activity I create several ProfitSplitter smart contracts with Solidity to accomplish the following across 3 levels of difficulty:

- **Level One** is an AssociateProfitSplitter contract. This will accept Ether into the contract and divide the Ether evenly among the associate level employees. This will allow the Human Resources department to pay employees quickly and efficiently.
- **Level Two** is a TieredProfitSplitter that will distribute different percentages of incoming Ether to employees at different tiers/levels. For example, the CEO gets paid 60%, CTO 25%, and Bob gets 15%.
- **Level Three** is a DeferredEquityPlan that models traditional company stock plans. This contract will automatically manage 1000 shares with an annual distribution of 250 over 4 years for a single employee.

Here is a snapshot of my Ganache workspace before transactions are conducted:

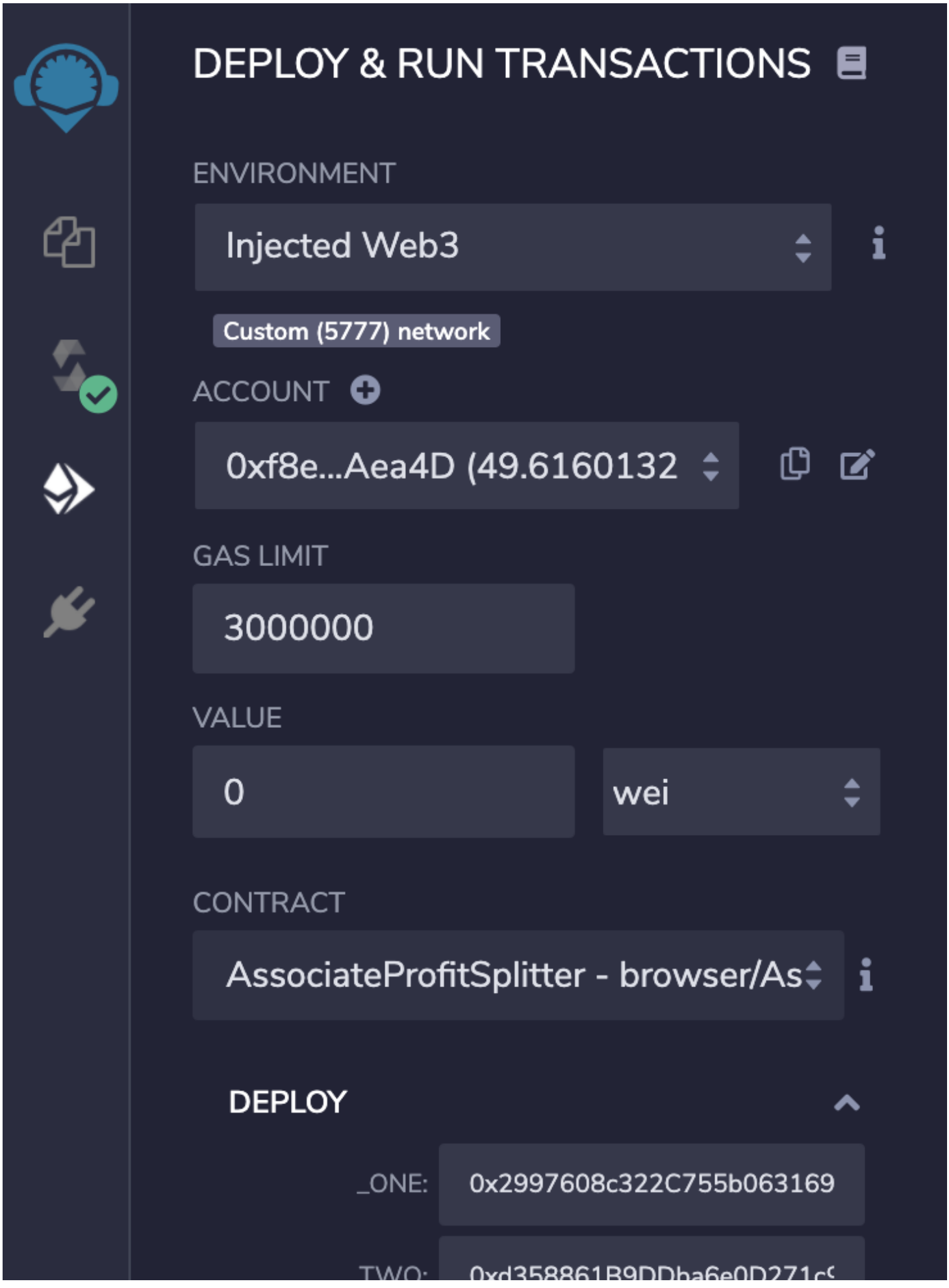


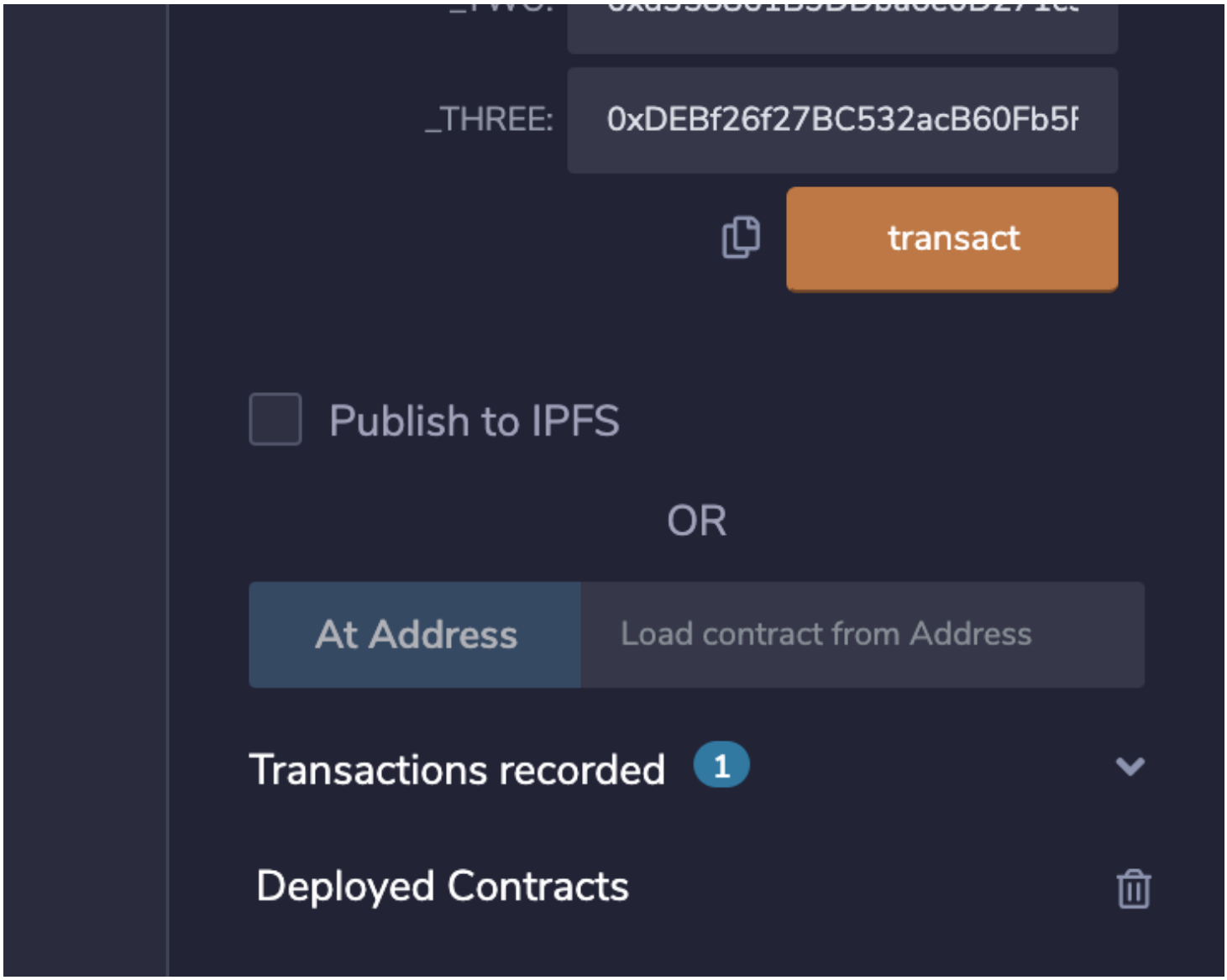
Level One: The AssociateProfitSplitter Contract

```
pragma solidity ^0.5.0;

// lvl 1: equal split
contract AssociateProfitSplitter {
    address payable employee_one;
    address payable employee_two;
    address payable employee_three;

    constructor(address payable _one, address payable _two, address payable _three) public {
        employee_one = _one;
        employee_two = _two;
        employee_three = _three;
    }

    function balance() public view returns(uint) {
        return address(this).balance;
    }

    function deposit() public payable {
        uint amount = msg.value / 3;

        employee_one.transfer(amount);
        employee_two.transfer(amount);
        employee_three.transfer(amount);

        msg.sender.transfer(msg.value - amount * 3);
    }

    function() external payable {
        deposit();
    }
}
```

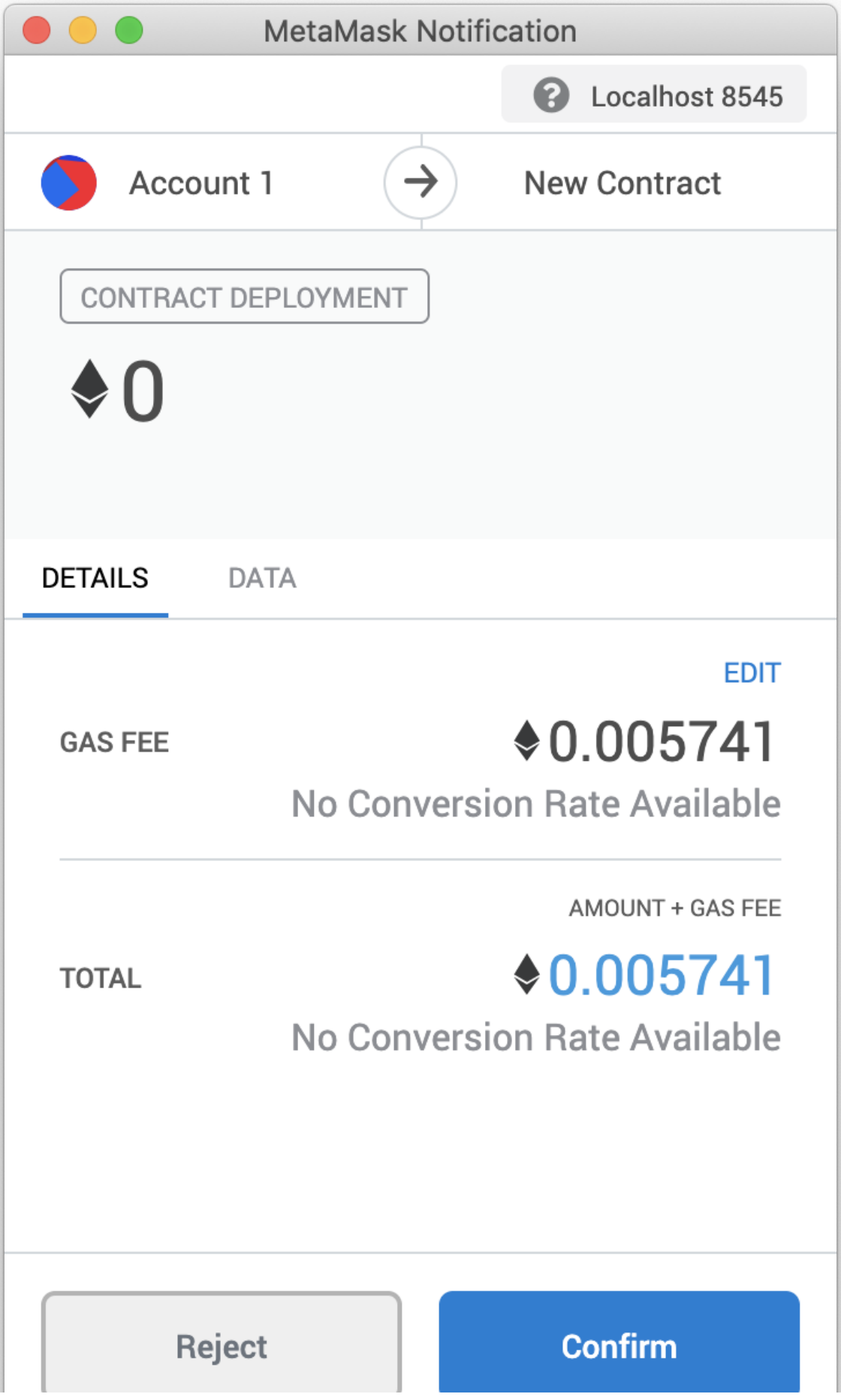
The code accepts Ether into the contract, divides the Ether evenly among the three recipient addresses, transfers the calculated amount to the three recipient addresses, and then transfers the remainder back to the sender address.

After successfully compiling the code, we switch the ENVIRONMENT to "Injected Web3" in the dropdown menu in Remix. We paste the three recipient addresses within the DEPLOY area and select "transact" (a value of 0 wei) to deploy our contract to the local network:

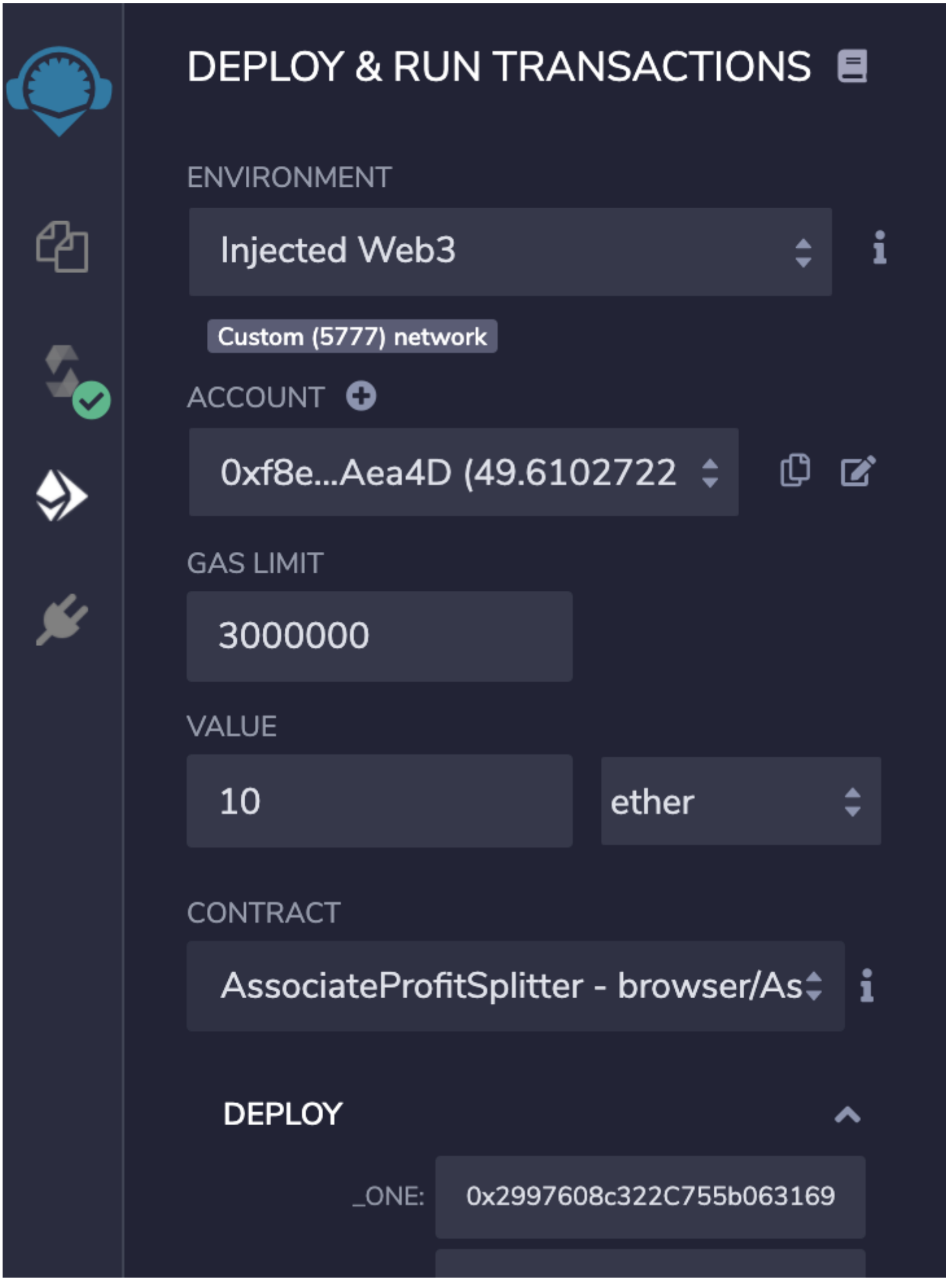

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Custom (5777) network
ACCOUNT
0xf8e...Aea4D (49.6160132
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
AssociateProfitSplitter - browser/As
DEPLOY
_ONE:
0x2997608c322C755b063169
TWO:

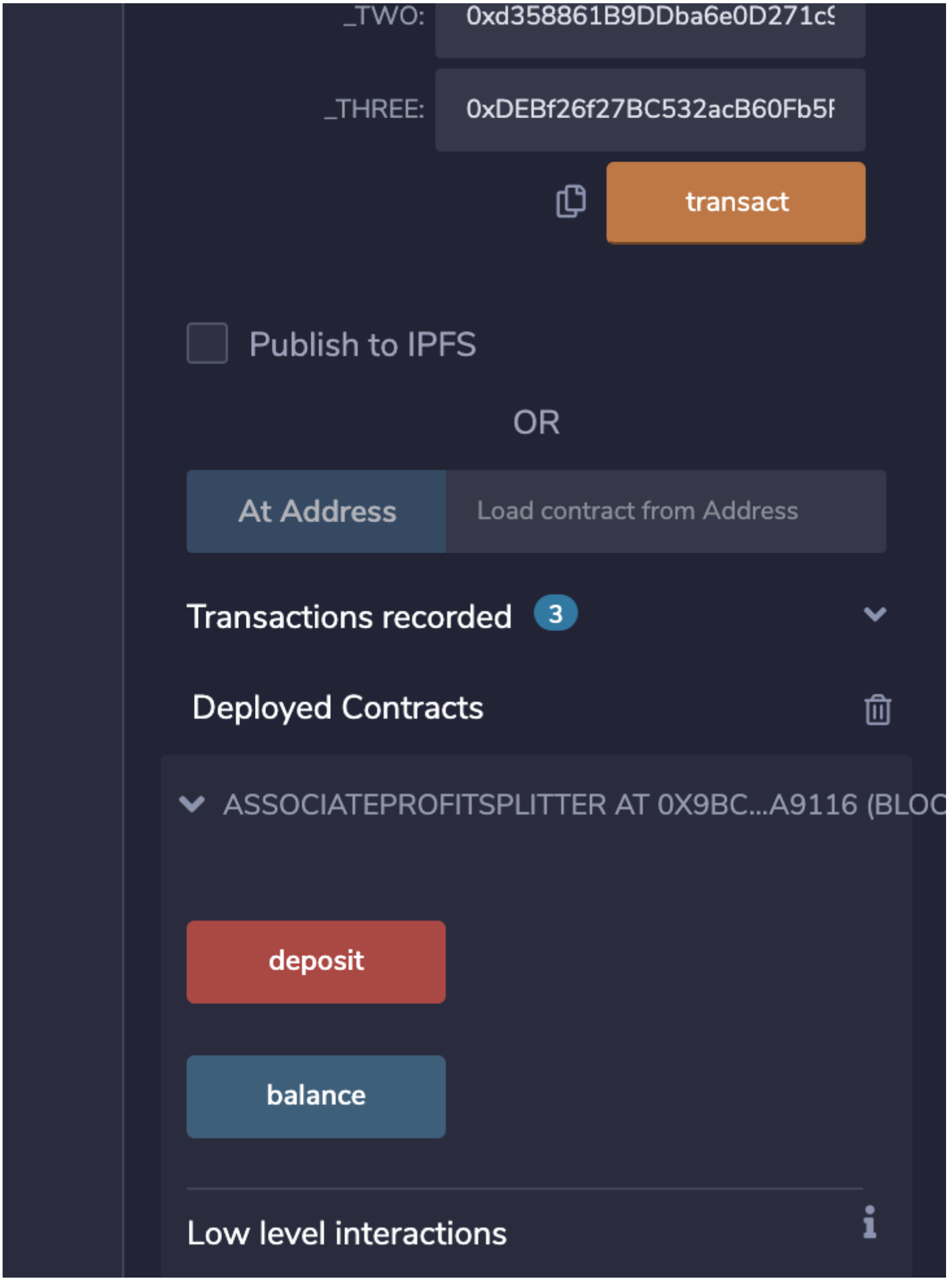


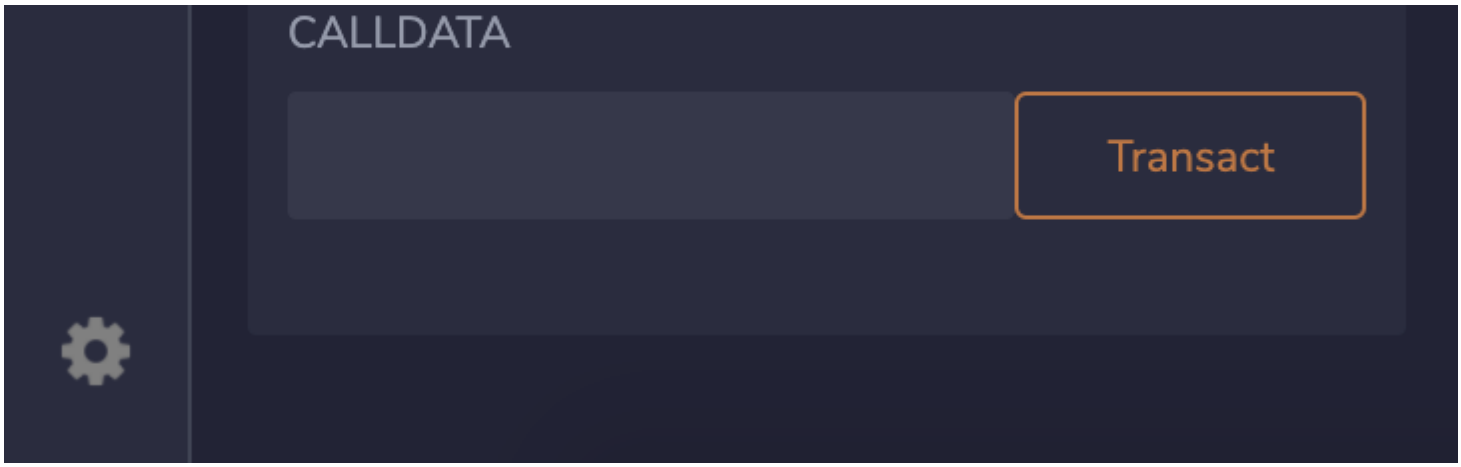
MetaMask pops up and we "Confirm" the contract deployment:


MetaMask Notification
Localhost 8545
Account 1
New Contract
CONTRACT DEPLOYMENT
0
DETAILS
DATA
EDIT
GAS FEE
0.005741
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
0.005741
No Conversion Rate Available
Reject
Confirm

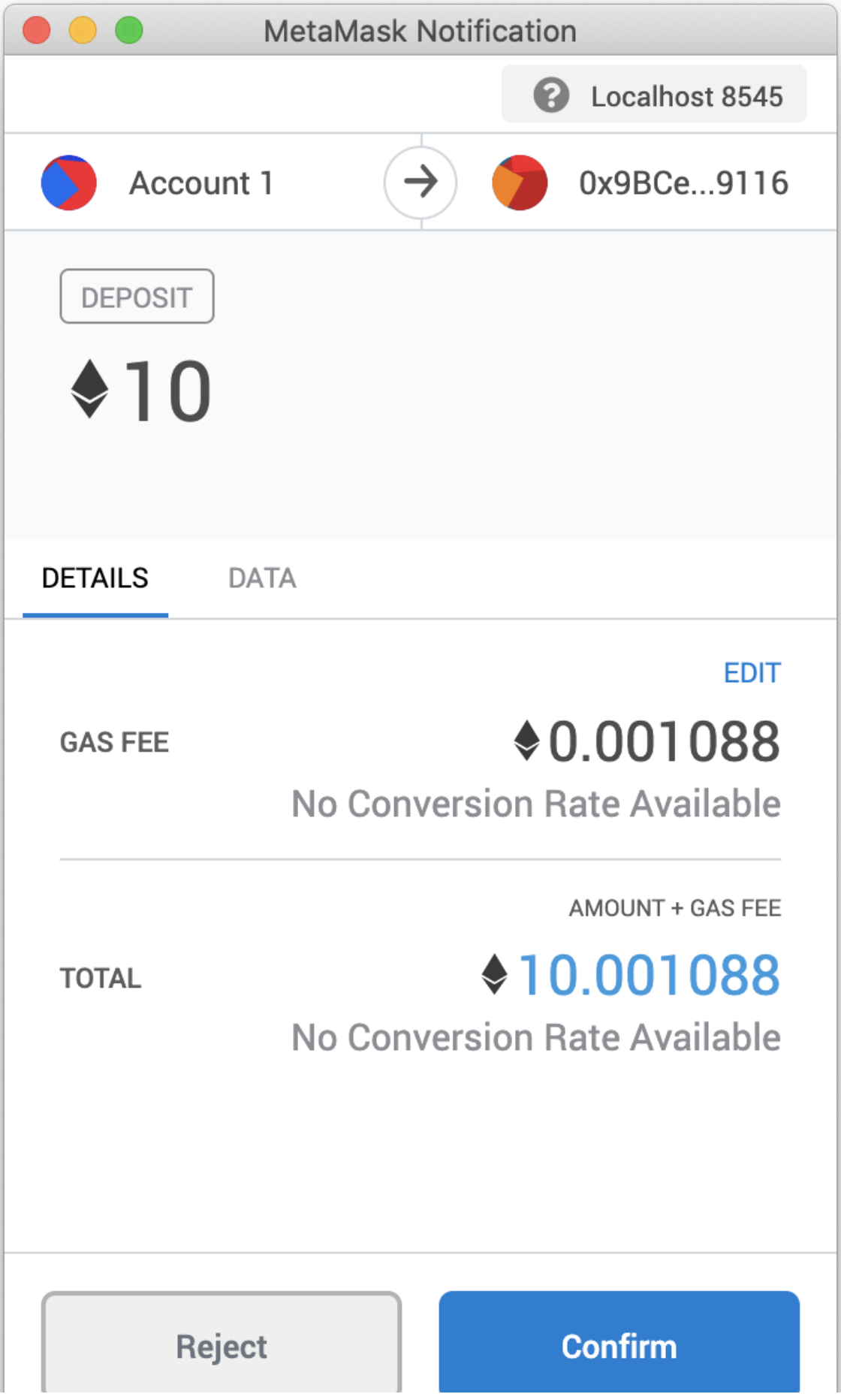
Now we can click an arrow beside "ASSOCIATEPROFITSPLITTER" in Remix to see our options to "deposit" or check "balance." We enter a VALUE of 10 ether to deposit:


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Custom (5777) network
ACCOUNT
0xf8e...Aea4D (49.6102722
GAS LIMIT
3000000
VALUE
10
ether
CONTRACT
AssociateProfitSplitter - browser/As
DEPLOY
_ONE:
0x2997608c322C755b063169

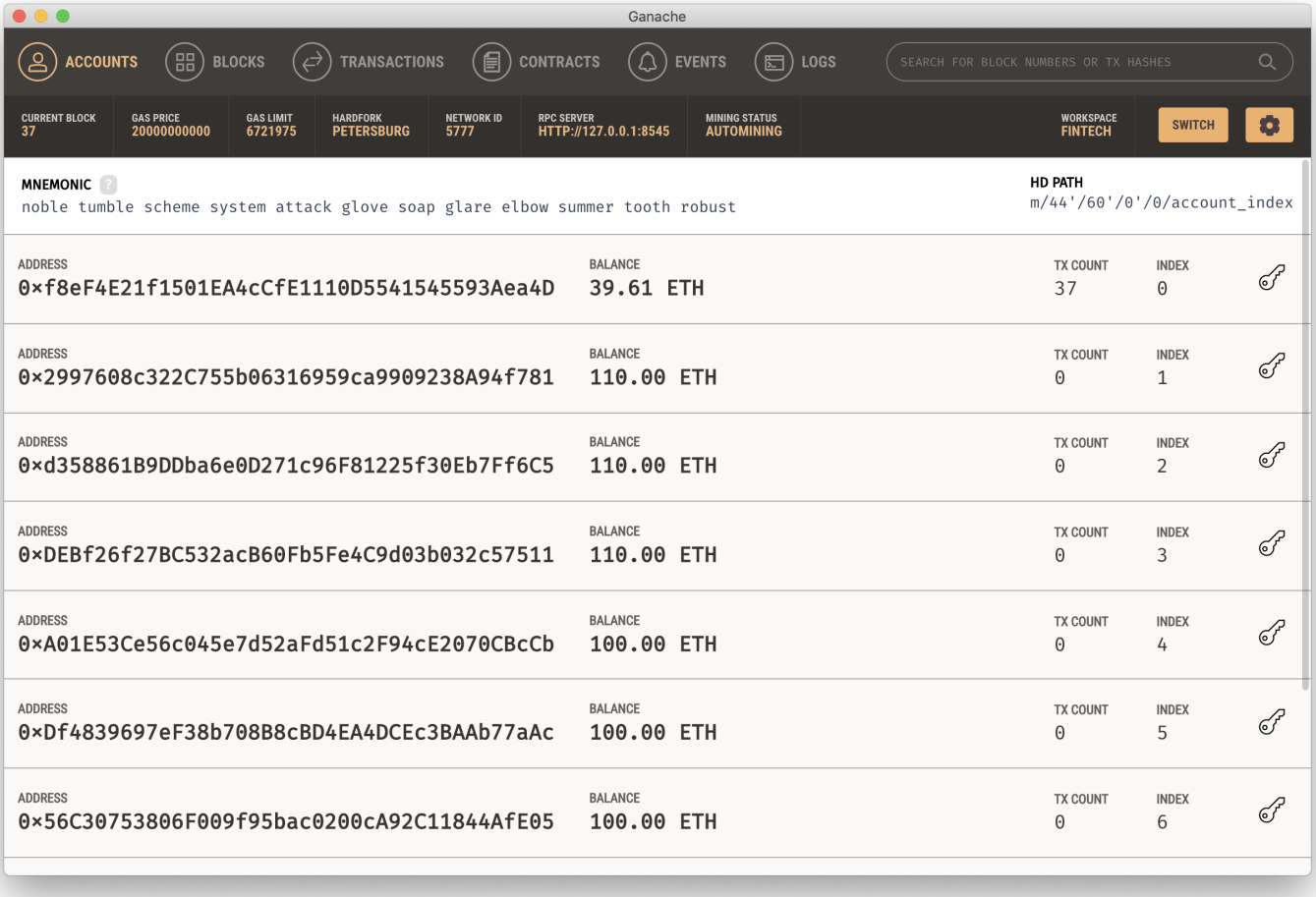

_TWO:
0xd358861B9DDba6e0D271c
_THREE:
0xDEBf26f27BC532acB60Fb5
transact
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
3
Deployed Contracts
ASSOCIATEPROFITSPLITTER AT 0X9BC...A9116 (BLOC
deposit
balance
Low level interactions



MetaMask again pops up and we confirm the transaction:


MetaMask Notification
Localhost 8545
Account 1
0x9BCe...9116
DEPOSIT
10
DETAILS
DATA
EDIT
GAS FEE
0.001088
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
10.001088
No Conversion Rate Available
Reject
Confirm

We can see in Ganache that the deposit worked!



Now we can point our MetaMask to the Kovan Test Network! We first check our balance in MetaMask on the Kovan Test Network:

Kovan Test Network

Connected

Account 1

0xf8eF...ea4D

0.9733 ETH

BUY **SEND**

Assets | Activity

Send ARCDMW

Sep 12 · To: 0x742c...263a

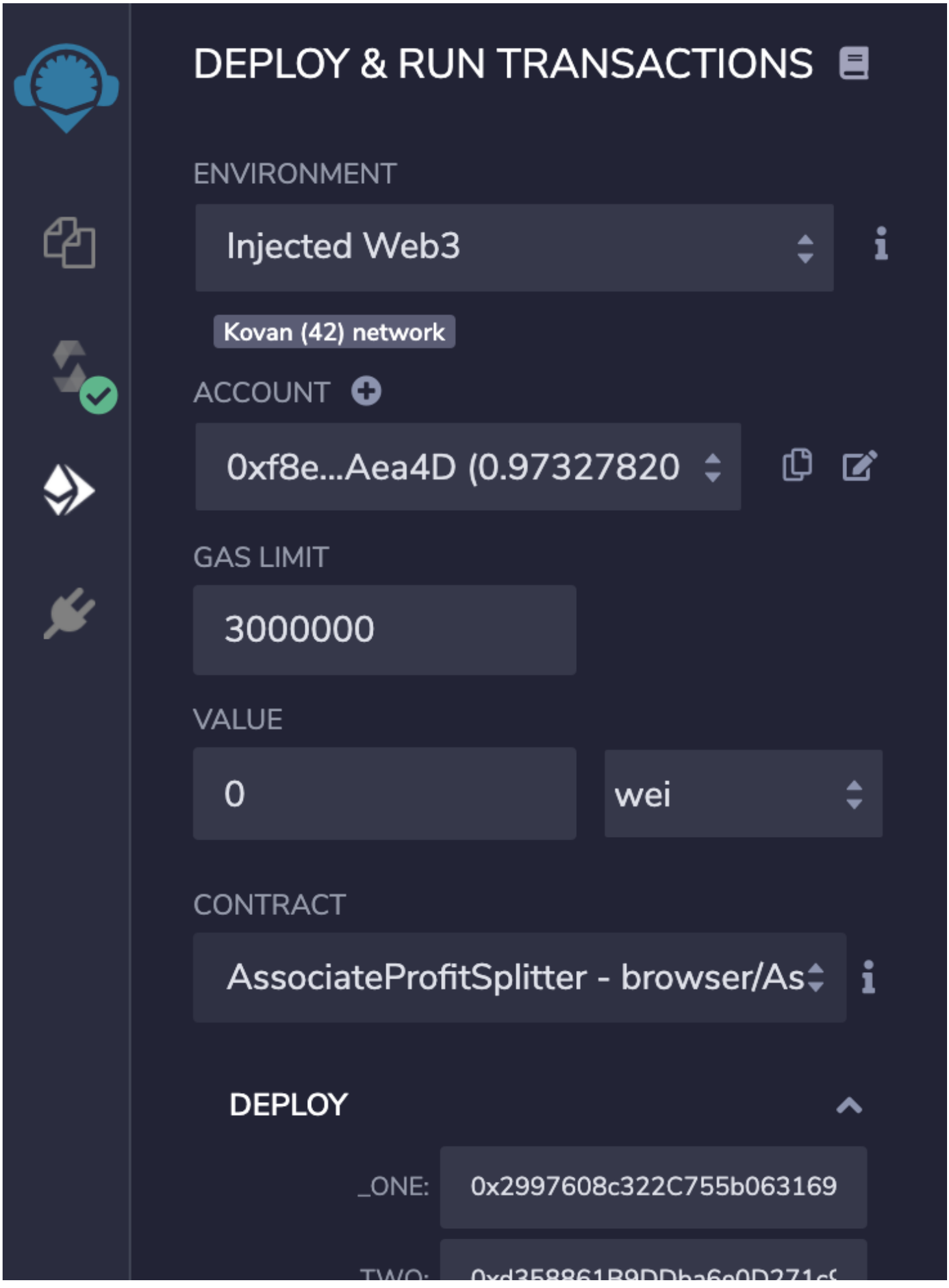
-1e-17 ARCDMW

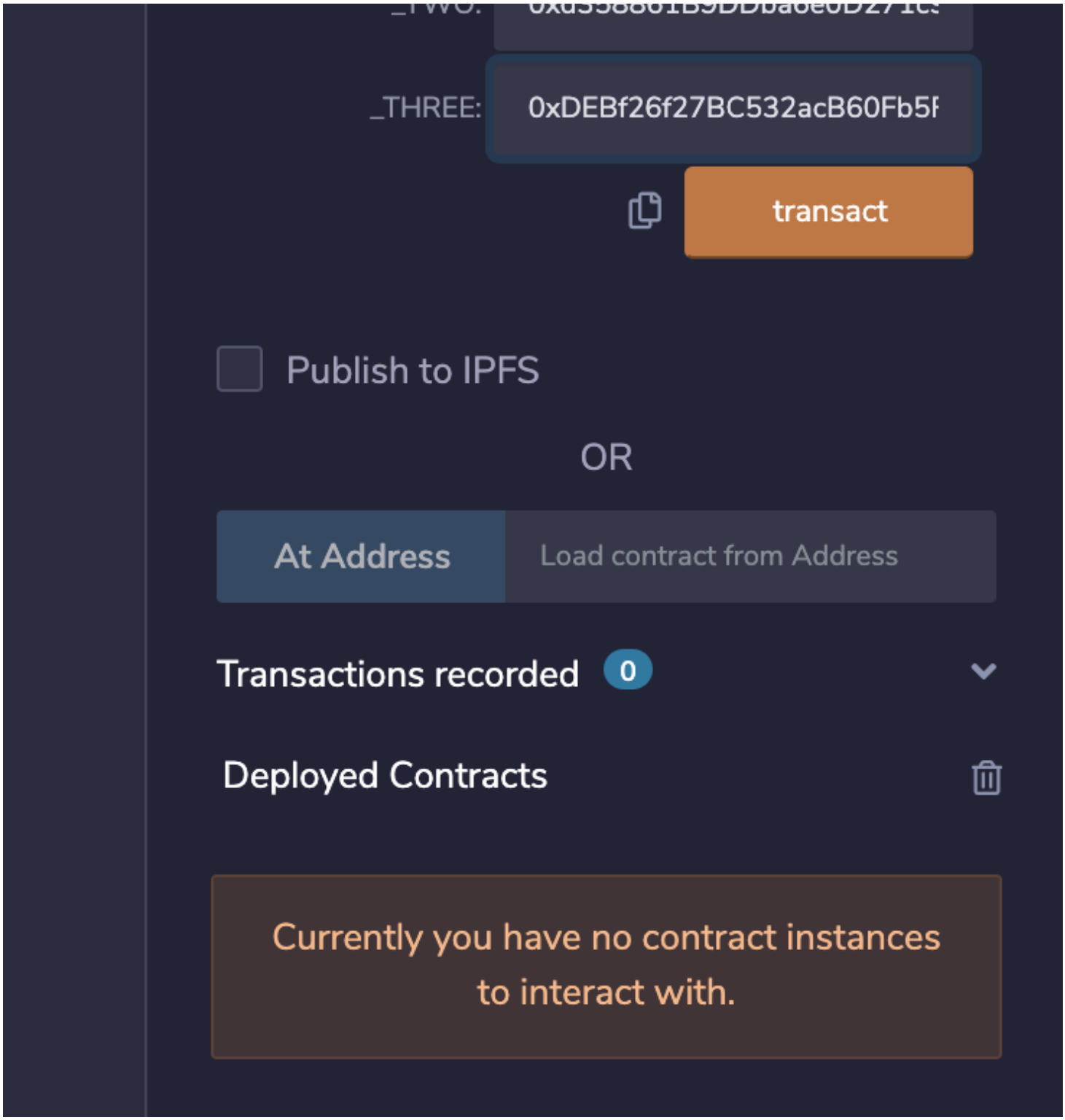
Send ARCDMW

Sep 12 · To: 0xa9c8...09a7

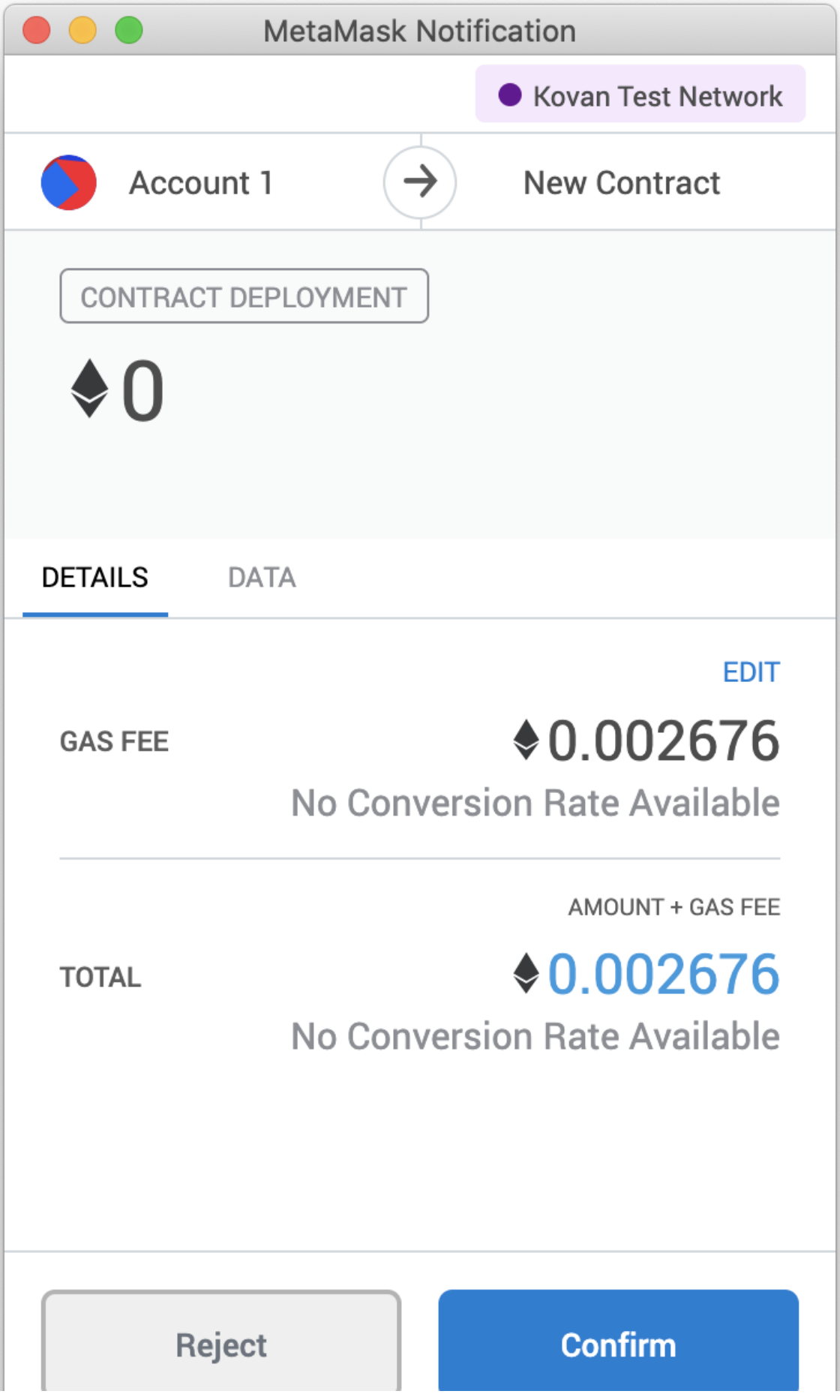
-1e-17
ARCDMW

We successfully compile our code. Then we change the ENVIRONMENT to Injected Web3 and populate the three recipient addresses under DEPLOY:

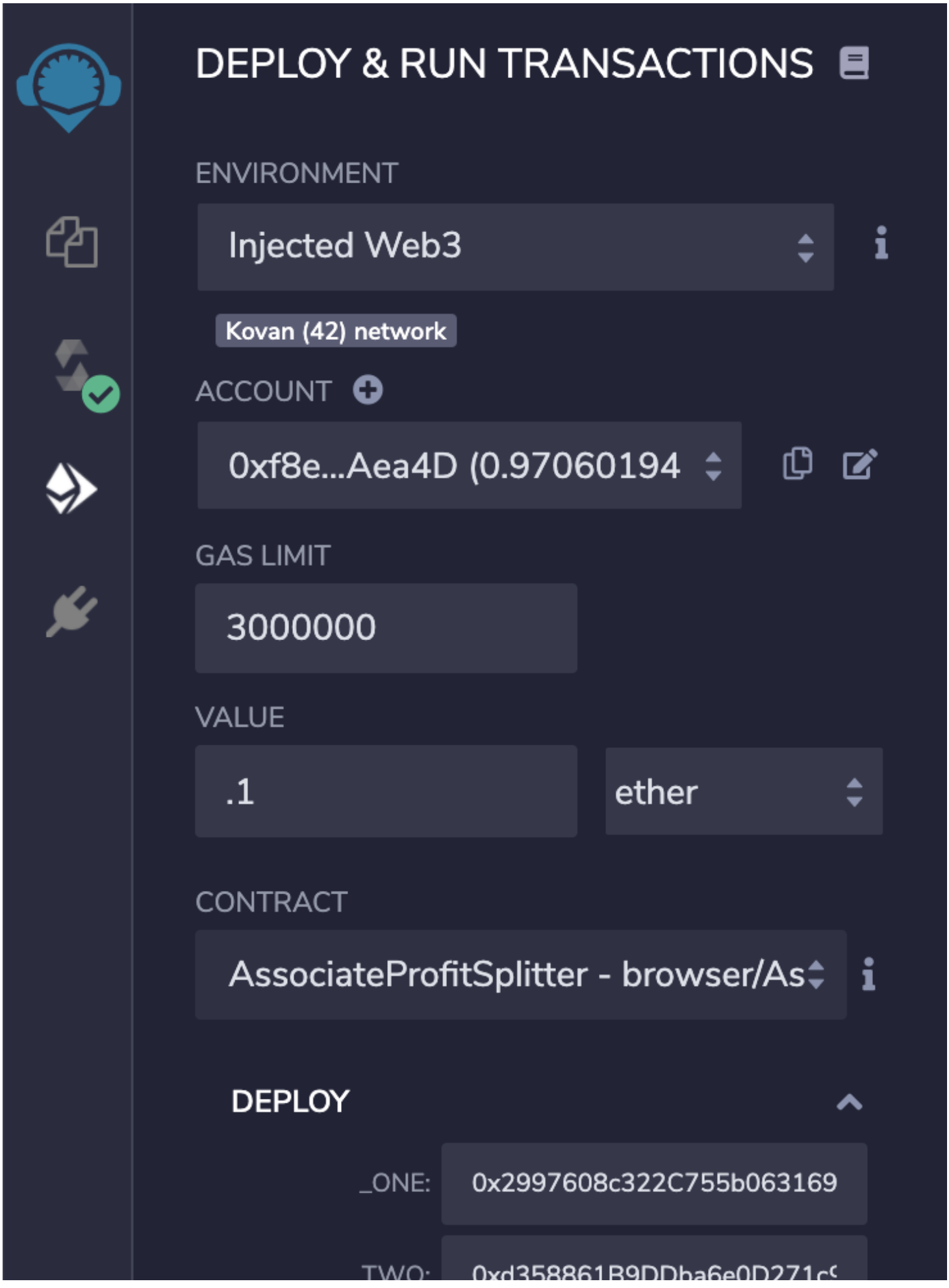

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Kovan (42) network
ACCOUNT
0xf8e...Aea4D (0.97327820
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
AssociateProfitSplitter - browser/As
DEPLOY
_ONE:
0x2997608c322C755b063169

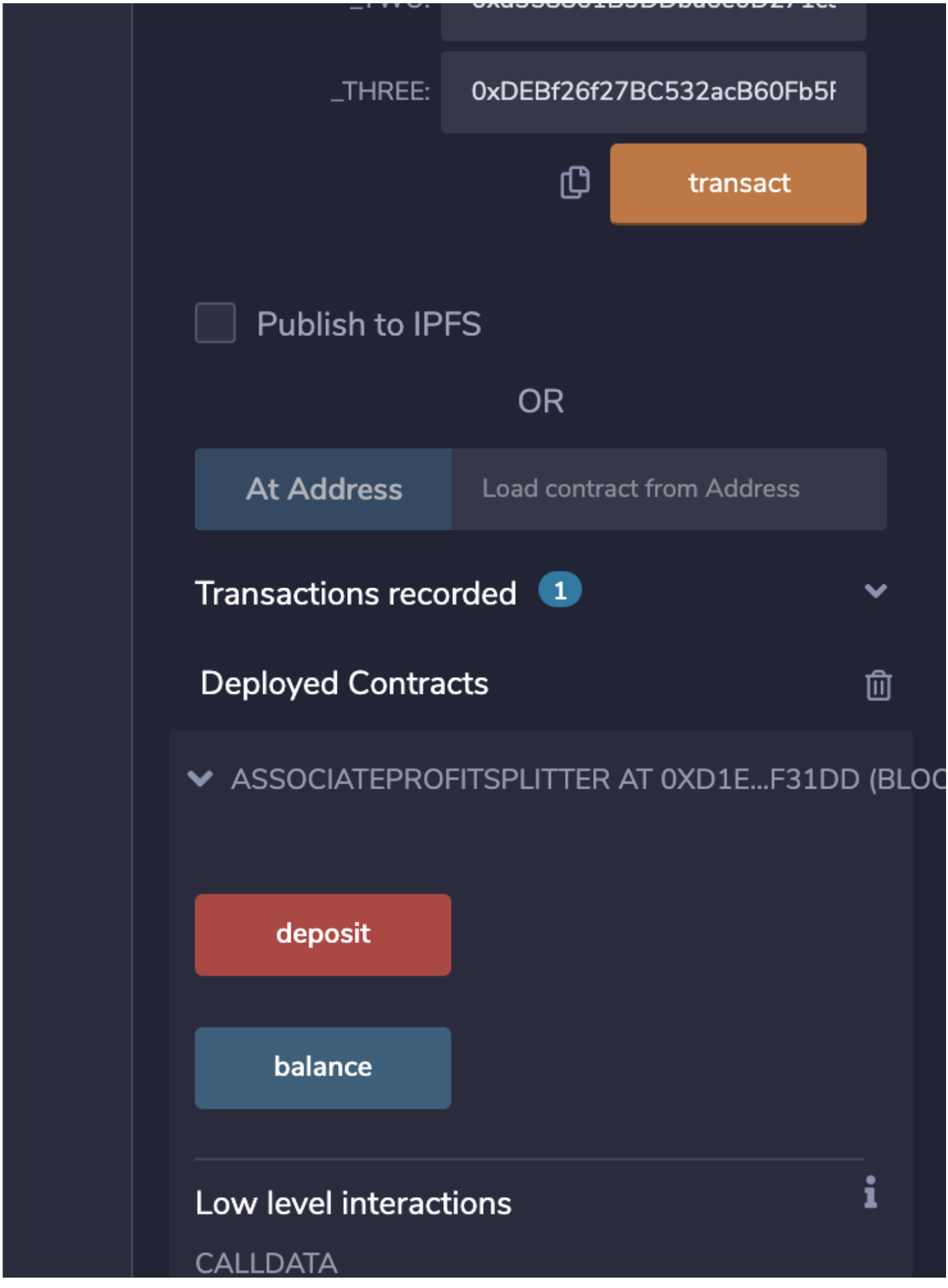


We confirm in MetMask:

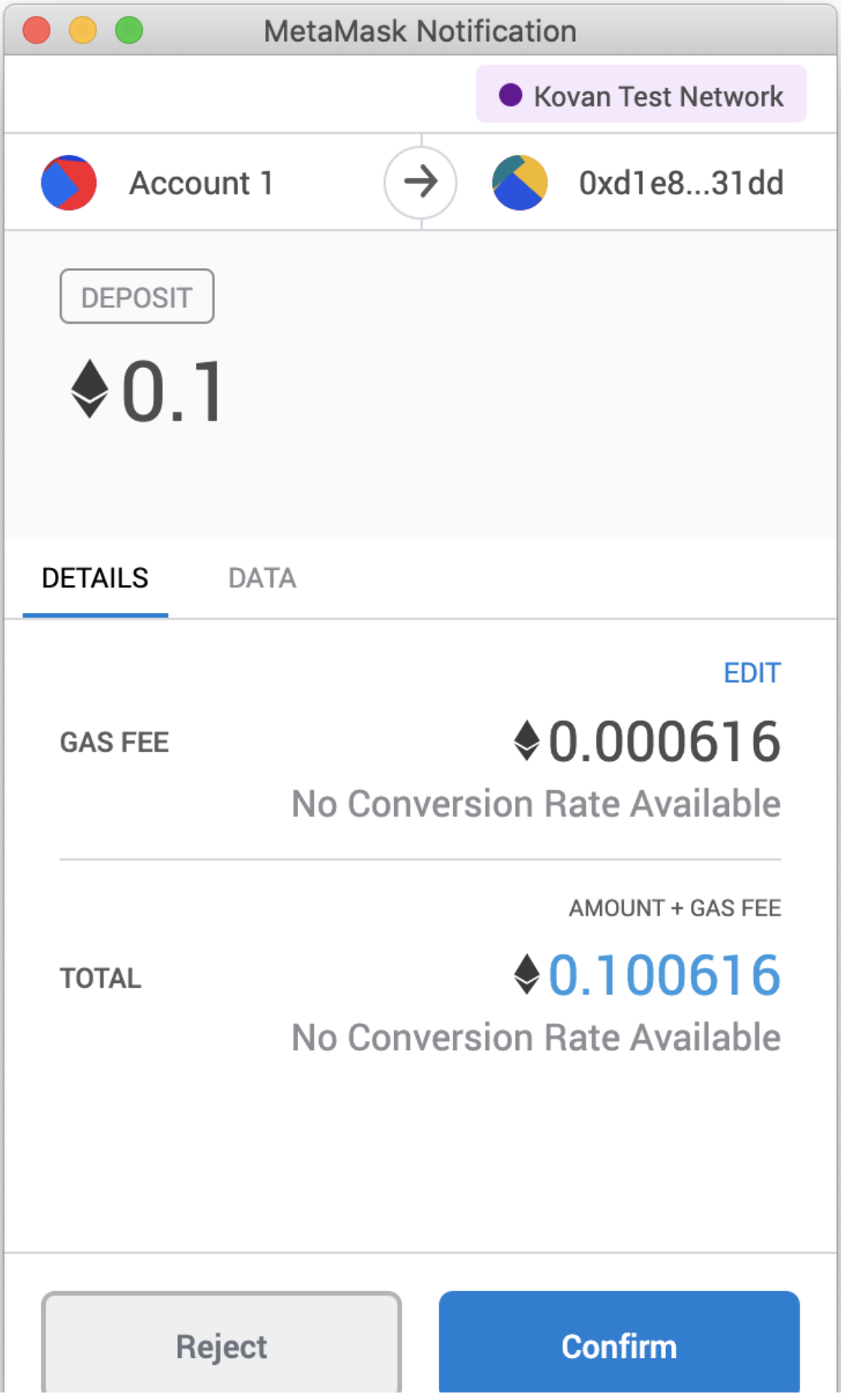

MetaMask Notification
Kovan Test Network
Account 1
New Contract
CONTRACT DEPLOYMENT
0
DETAILS
DATA
EDIT
GAS FEE
0.002676
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
0.002676
No Conversion Rate Available
Reject
Confirm

This generates our option buttons to "deposit" or check "balance" in the lower left. We choose to "deposit" a VALUE of 0.1 ether:

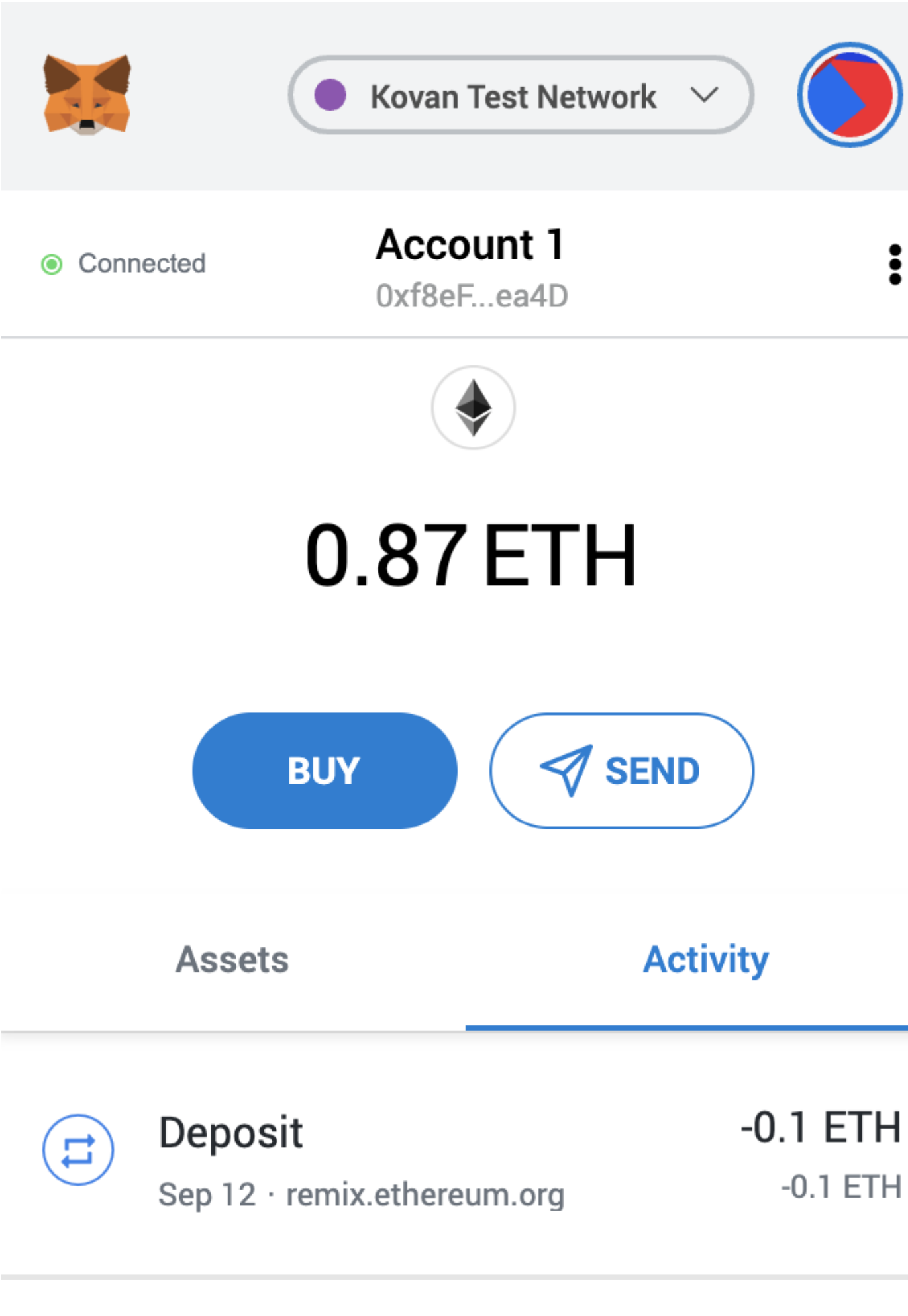

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Kovan (42) network
ACCOUNT
0xf8e...Aea4D (0.97060194
GAS LIMIT
3000000
VALUE
.1
ether
CONTRACT
AssociateProfitSplitter - browser/As
DEPLOY
_ONE:
0x2997608c322C755b063169
TWO:

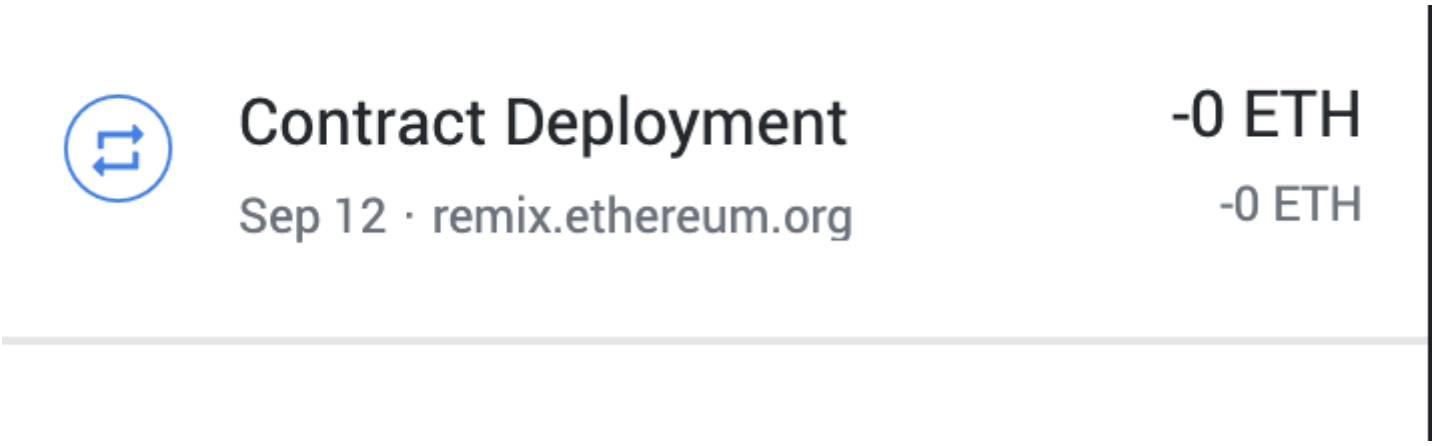

_THREE:
0xDEBf26f27BC532acB60Fb5f
transact
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
1
Deployed Contracts
ASSOCIATEPROFITSPLITTER AT 0XD1E...F31DD (BLOC
deposit
balance
Low level interactions
CALLDATA

We confirm our deposit in MetaMask:

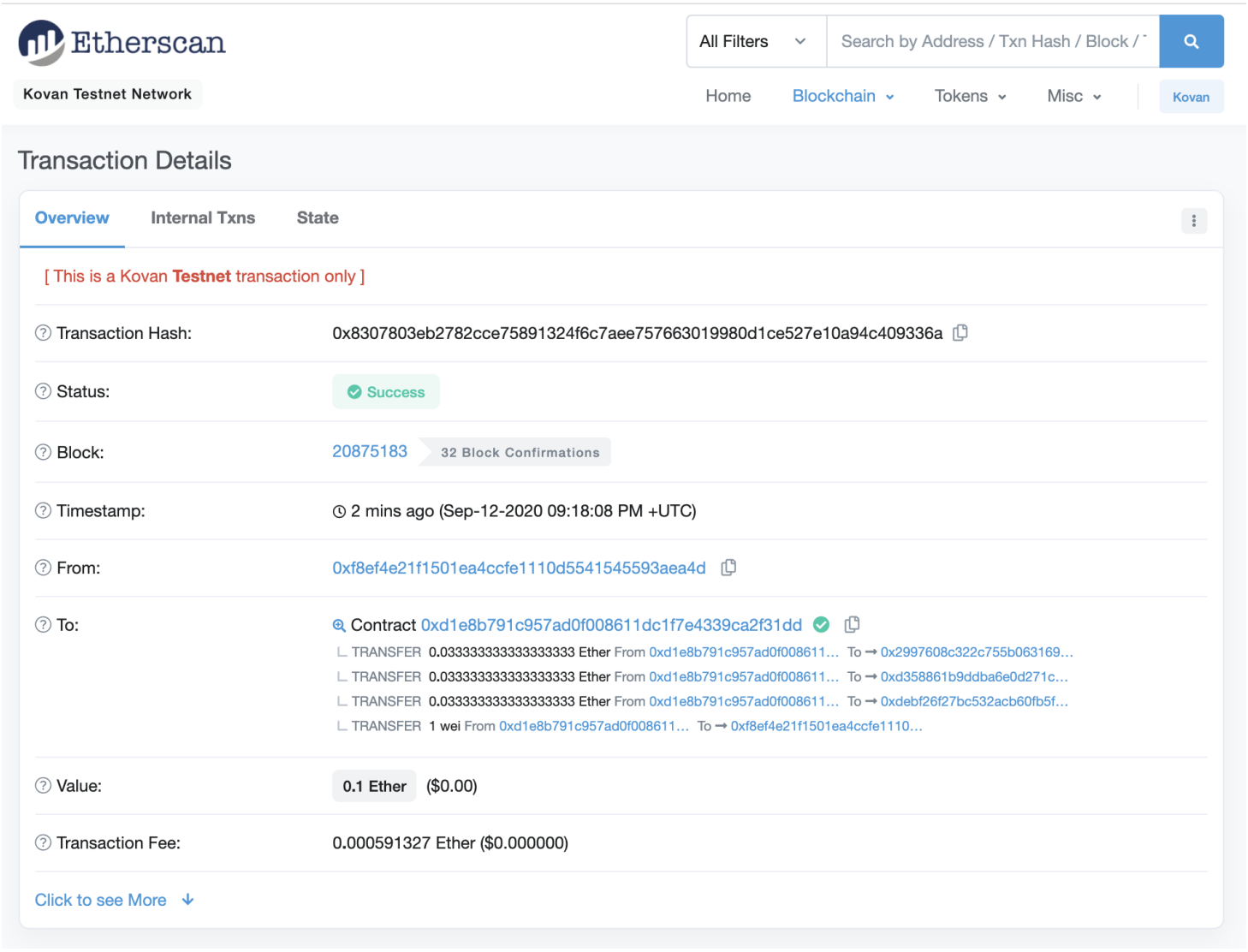

MetaMask Notification
Kovan Test Network
Account 1
0xd1e8...31dd
DEPOSIT
0.1
DETAILS
DATA
EDIT
GAS FEE
0.000616
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
0.100616
No Conversion Rate Available
Reject
Confirm

We can see that the deposit worked and our account balance has decreased:


Kovan Test Network
Connected
Account 1
0xf8eF...ea4D
0.87 ETH
BUY
SEND
Assets
Activity
Deposit
Sep 12 · remix.ethereum.org
-0.1 ETH
-0.1 ETH



We double-check etherscan to determine that the deposit was successfully routed into the three recipient accounts, as well as the remainder of 1 wei that goes back into the sender account:



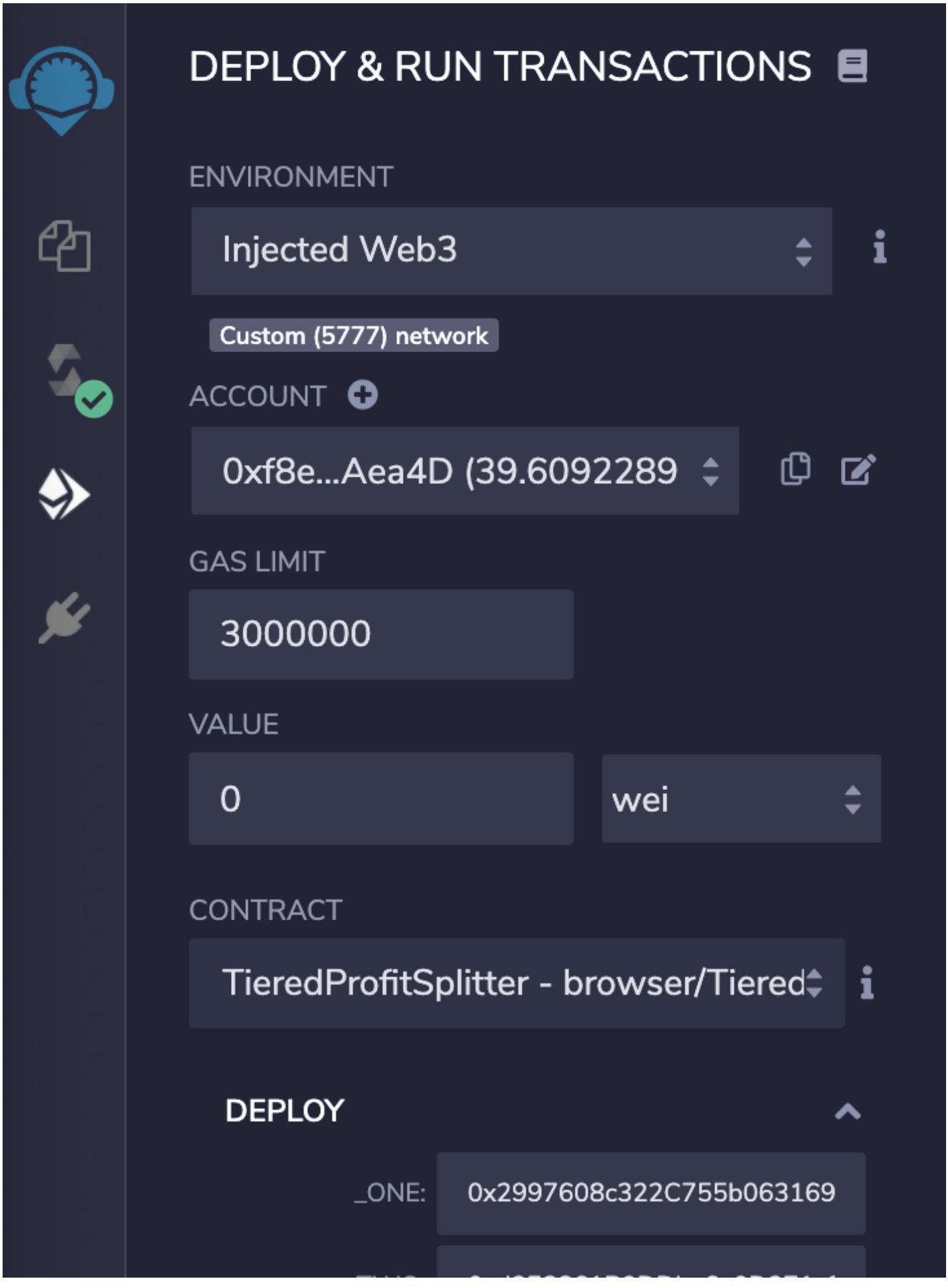
The testnet address for others to be able to send to is: 0xF40870a74528482E4ba15a8450687b0ee778fe68.

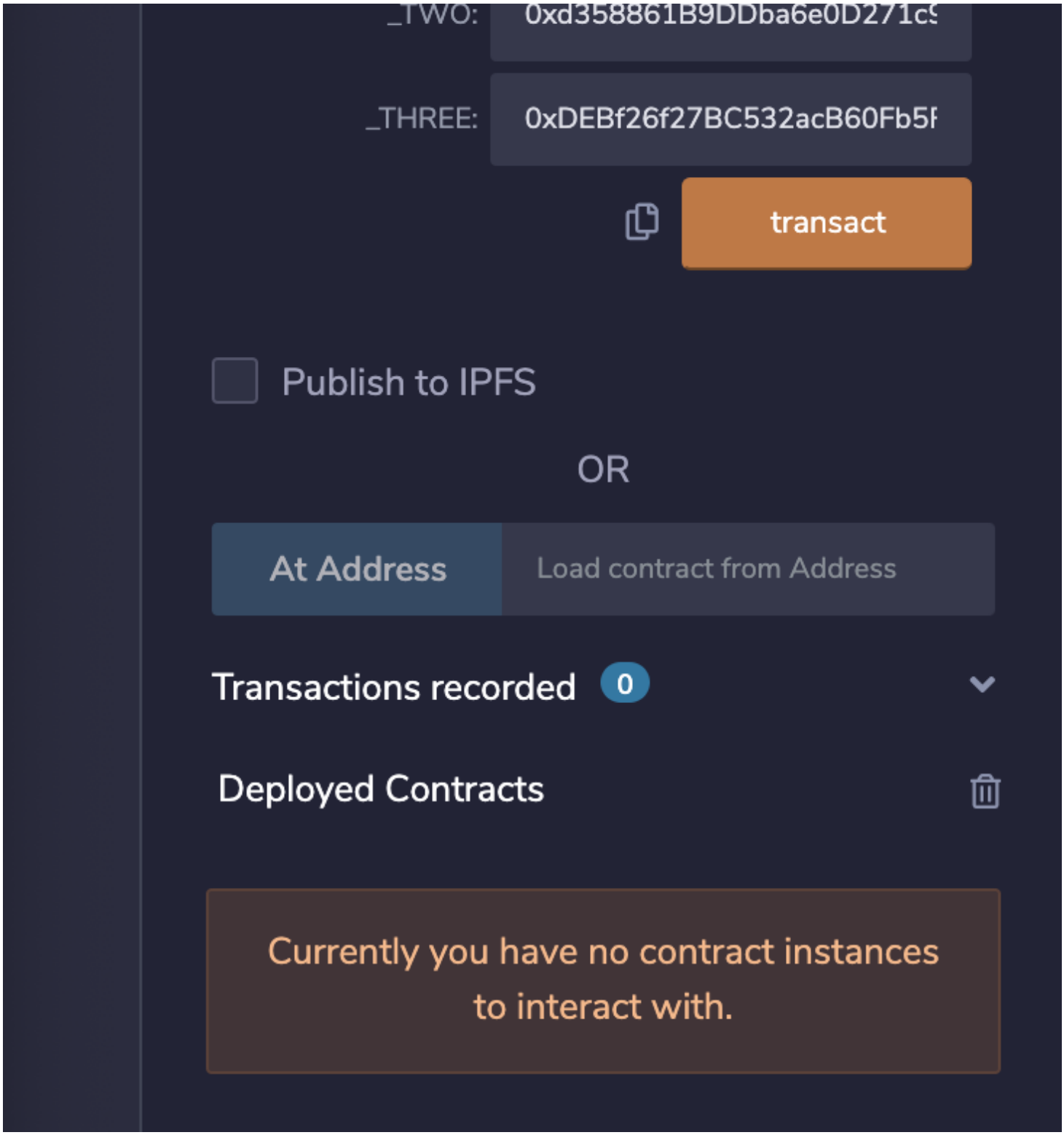
Level Two: The TieredProfitSplitter Contract

```
pragma solidity ^0.5.0;

// lvl 2: tiered split
contract TieredProfitSplitter {
    address payable employee_one; // ceo
    address payable employee_two; // cto
    address payable employee_three; // bob

    constructor(address payable _one, address payable _two, address payable _three) public {
        employee_one = _one;
        employee_two = _two;
        employee_three = _three;
    }

    // Should always return 0! Use this to test your `deposit` function's logic
    function balance() public view returns(uint) {
        return address(this).balance;
    }

    function deposit() public payable {
        uint points = msg.value / 100; // Calculates rudimentary percentage by dividing msg.value into
        uint total;
        uint amount;

        // @TODO: Calculate and transfer the distribution percentage
        // Step 1: Set amount to equal `points` * the number of percentage points for this employee
        // Step 2: Add the `amount` to `total` to keep a running total
        // Step 3: Transfer the `amount` to the employee

        amount = points * 60;
        total += amount;
        employee_one.transfer(amount);

        // @TODO: Repeat the previous steps for `employee_two` and `employee_three`

        amount = points * 25;
        total += amount;
        employee_two.transfer(amount);

        amount = points * 15;
        total += amount;
        employee_three.transfer(amount);

        employee_one.transfer(msg.value - total); // ceo gets the remaining wei
    }

    function() external payable {
        deposit();
    }
}
```

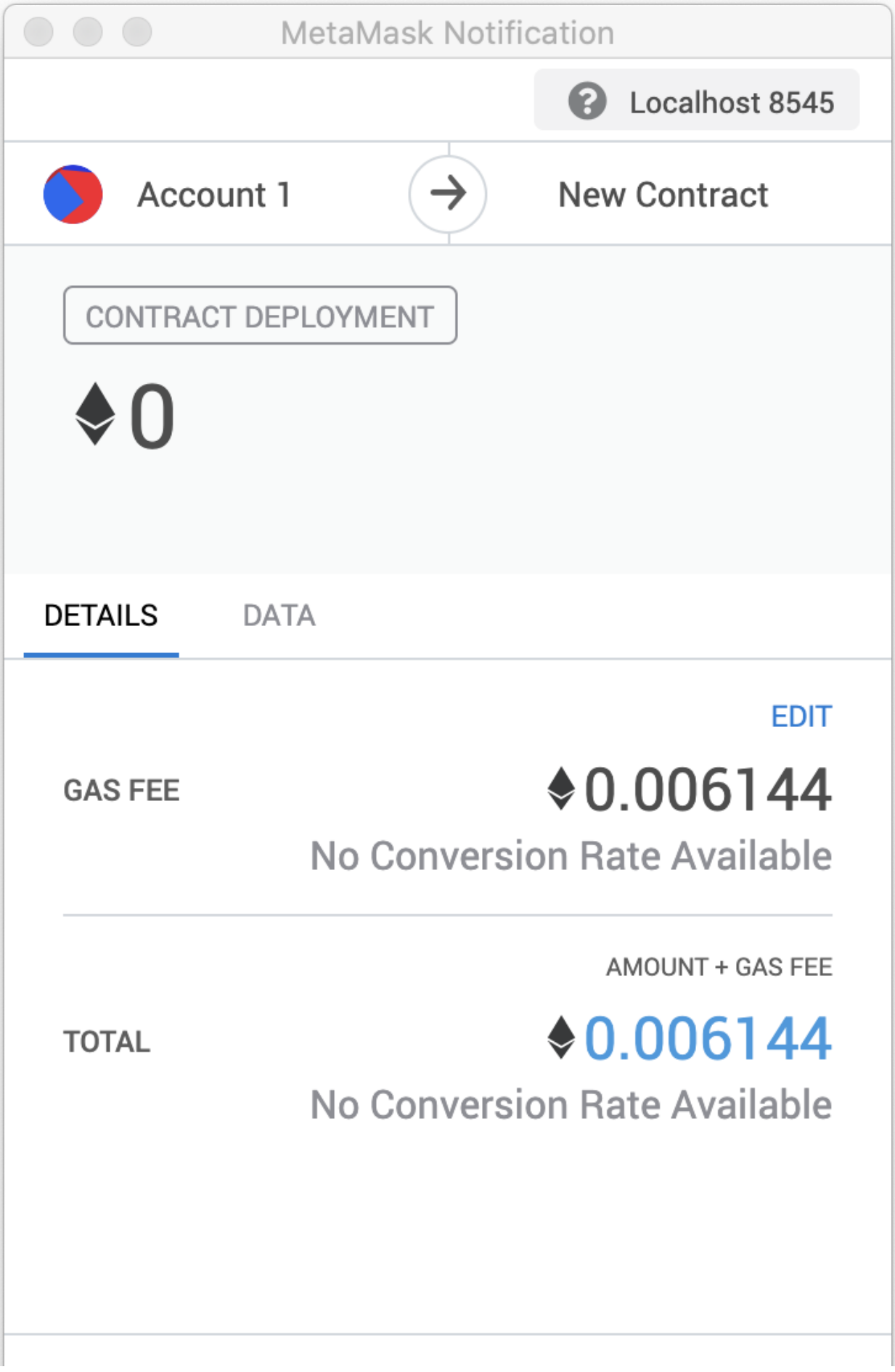
The code calculates one percentage of the transferred amount and then splits it among three recipient accounts with different tiers of compensation. It also keeps a running total of the amounts transferred to each employee and then transfers any remainder to the employee with the highest percentage by subtracting the running total from msg.value.

We switch back to our Localhost 8545 network and successfully compile the code, then switch the ENVIRONMENT to Injected Web3. We enter the three recipient addresses and click "transact" with a VALUE of 0 wei:

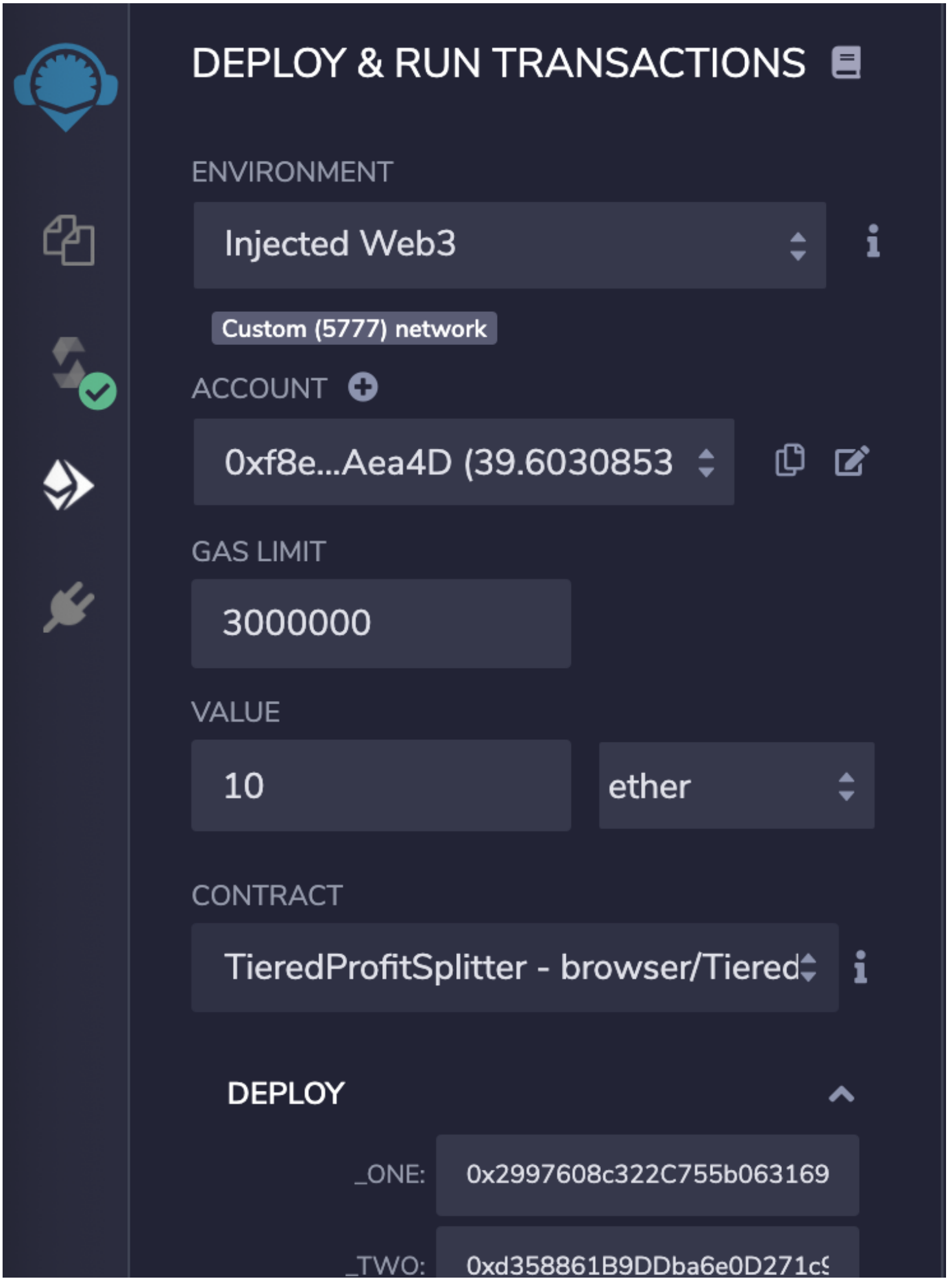

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Custom (5777) network
ACCOUNT
0xf8e...Aea4D (39.6092289
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
TieredProfitSplitter - browser/Tiered
DEPLOY
_ONE:
0x2997608c322C755b063169

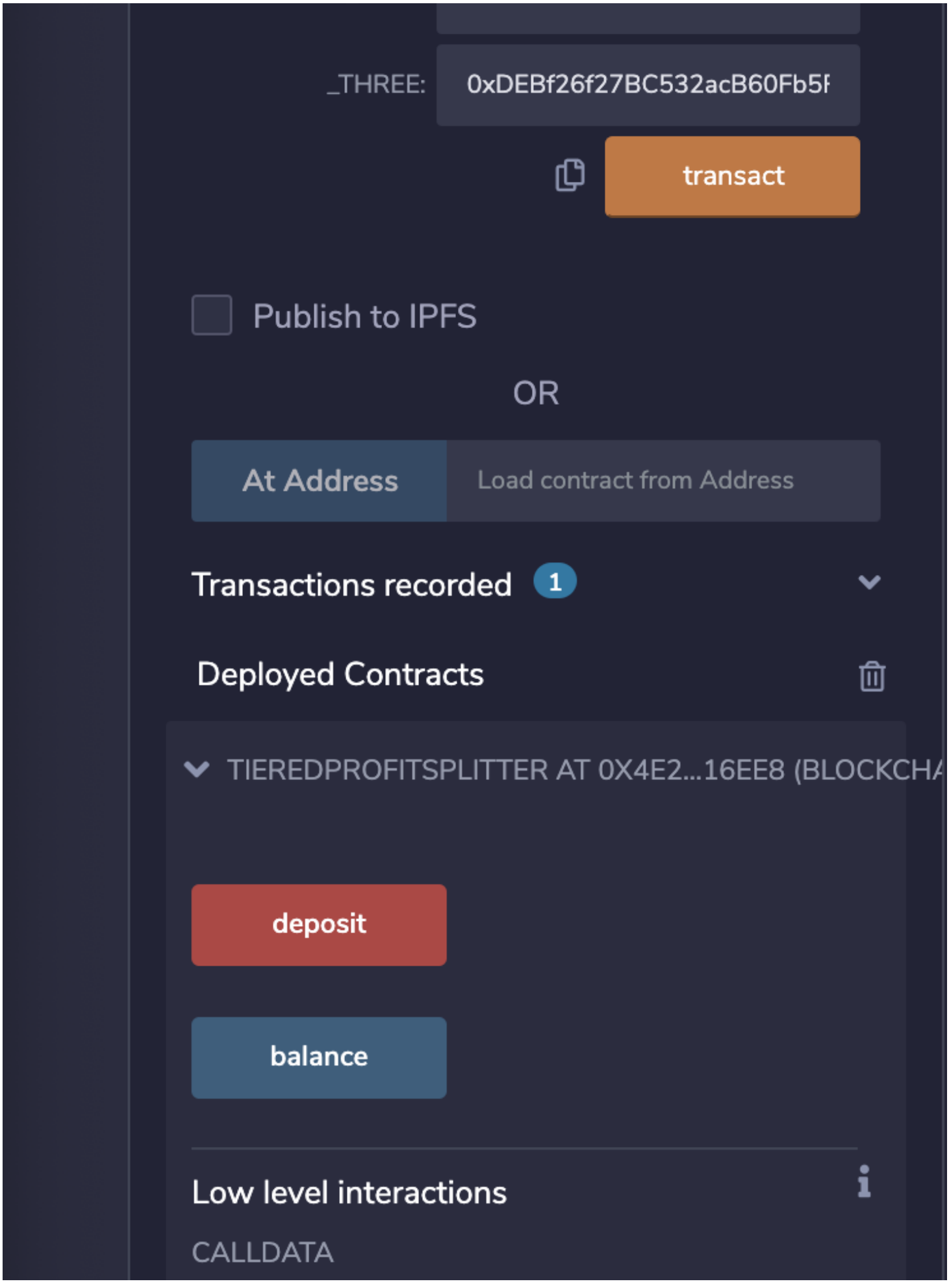


MetaMask pops up and we Confirm:

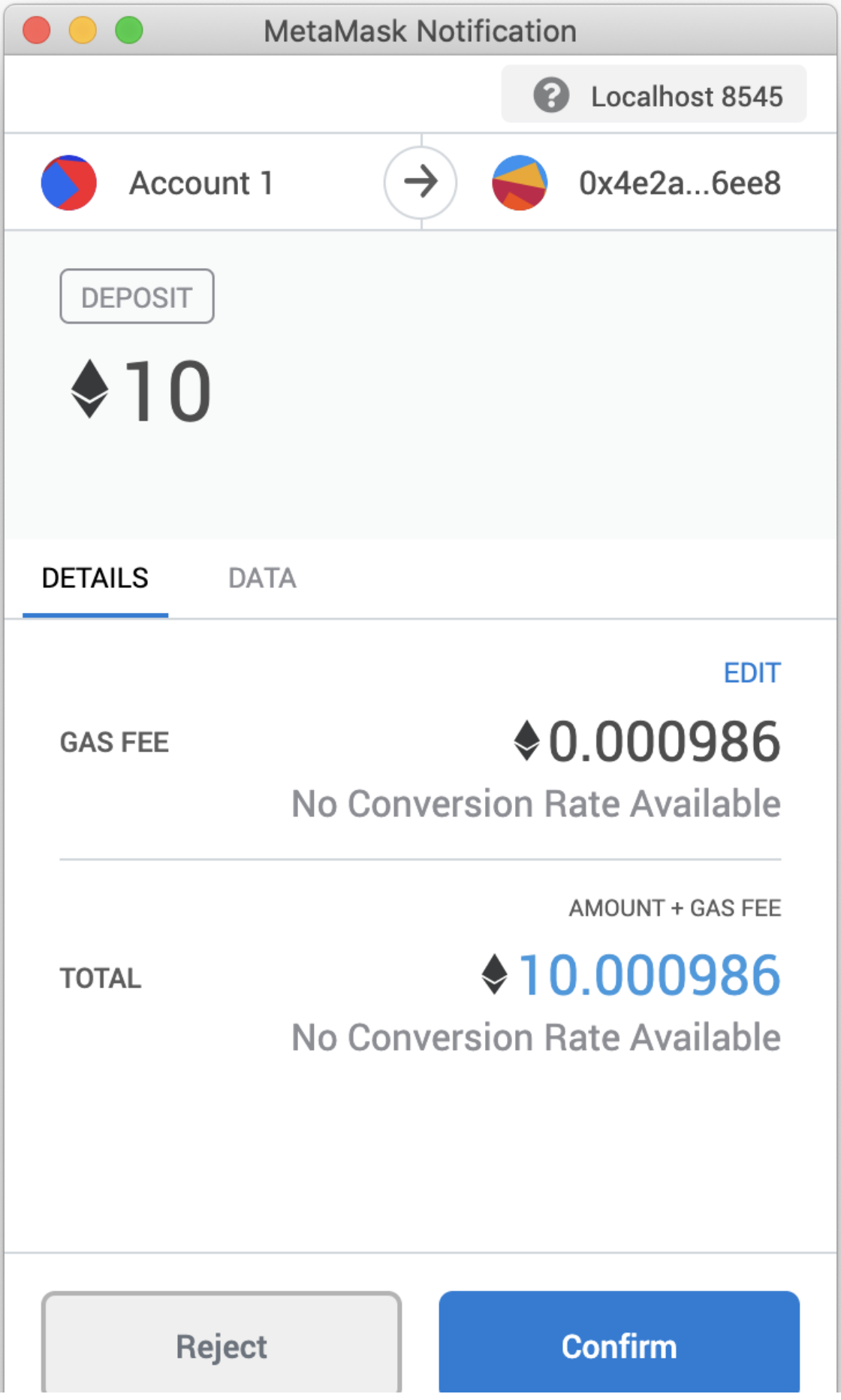

MetaMask Notification
Localhost 8545
Account 1
New Contract
CONTRACT DEPLOYMENT
0
DETAILS
DATA
EDIT
GAS FEE
0.006144
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
0.006144
No Conversion Rate Available

Now we can view the dropdown "deposit" and "balance" buttons in the lower left of Remix. We choose to transfer a VALUE of 10 ether and click "deposit":

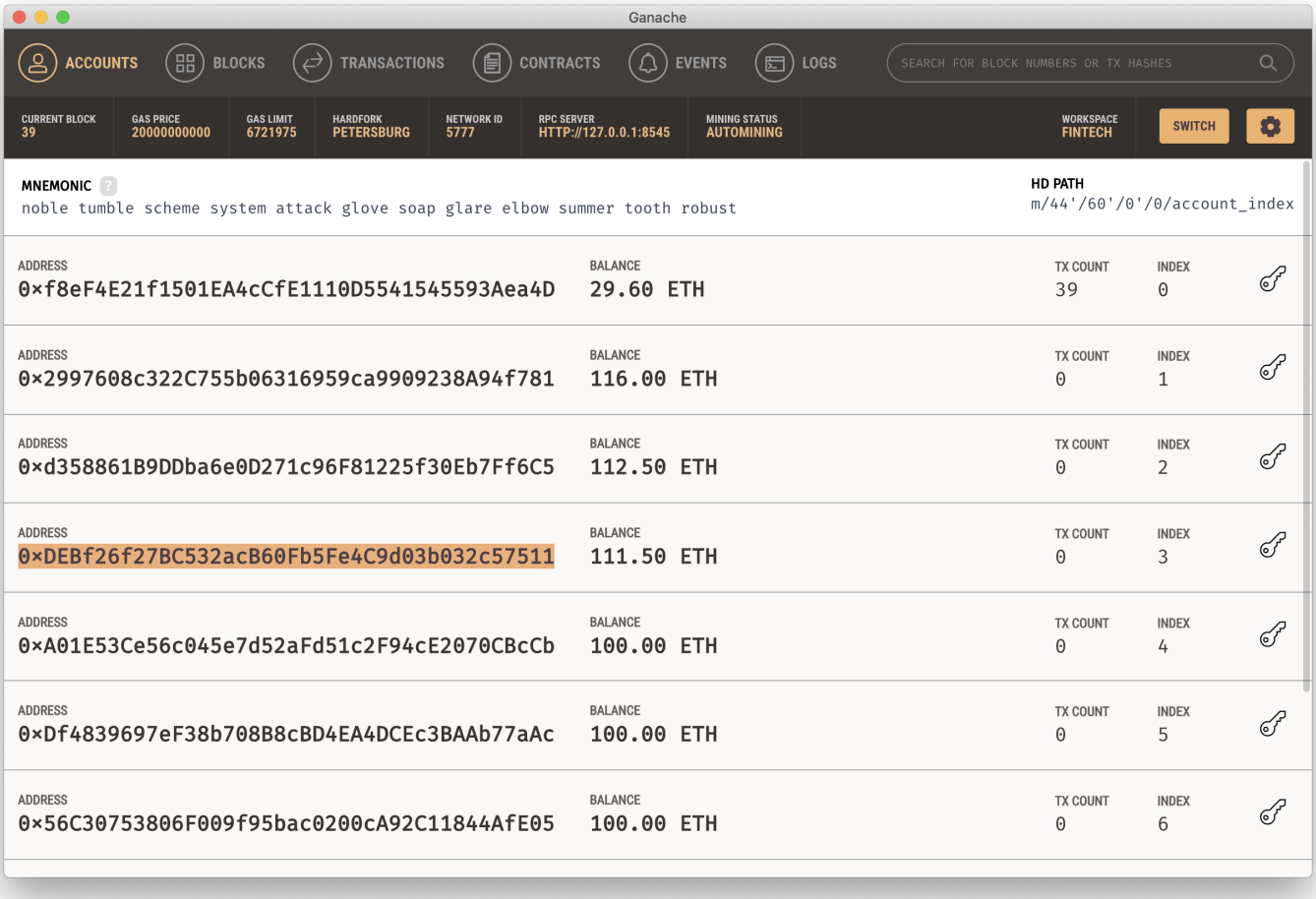

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Custom (5777) network
ACCOUNT
0xf8e...Aea4D (39.6030853
GAS LIMIT
3000000
VALUE
10
ether
CONTRACT
TieredProfitSplitter - browser/Tiered
DEPLOY
_ONE:
0x2997608c322C755b063169
_TWO:
0xd358861B9DDba6e0D271c


_THREE:
0xDEBf26f27BC532acB60Fb5f
transact
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
1
Deployed Contracts
TIEREDPROFITSPLITTER AT 0X4E2...16EE8 (BLOCKCHA
deposit
balance
Low level interactions
CALLDATA

MetaMask pops up and we confirm the deposit:


MetaMask Notification
Localhost 8545
Account 1
0x4e2a...6ee8
DEPOSIT
10
DETAILS
DATA
EDIT
GAS FEE
0.000986
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
10.000986
No Conversion Rate Available
Reject
Confirm

We are able to see in Ganache that our deposits were successful and our account balances have updated!



Now we switch back to the Kovan Test Network in MetaMask:

Kovan Test Network

Connected

Account 1

0xf8eF...ea4D

0.87 ETH

BUY

SEND

Assets

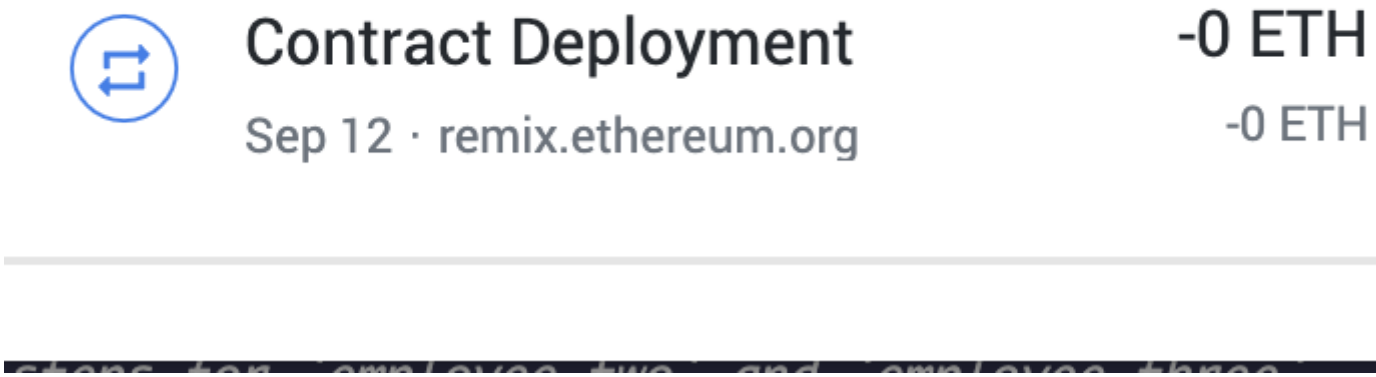
Activity

Deposit

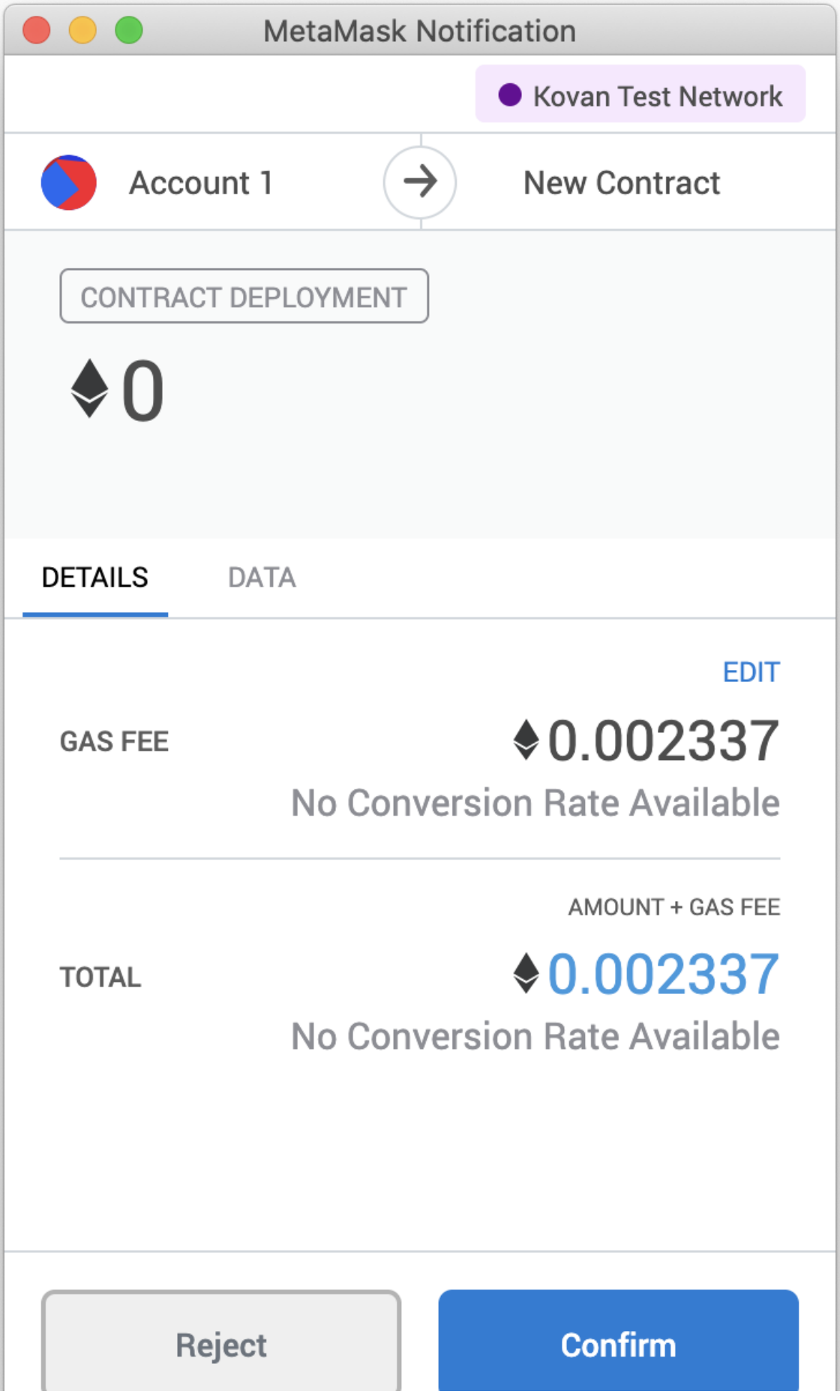
Sep 12 · remix.ethereum.org

-0.1 ETH

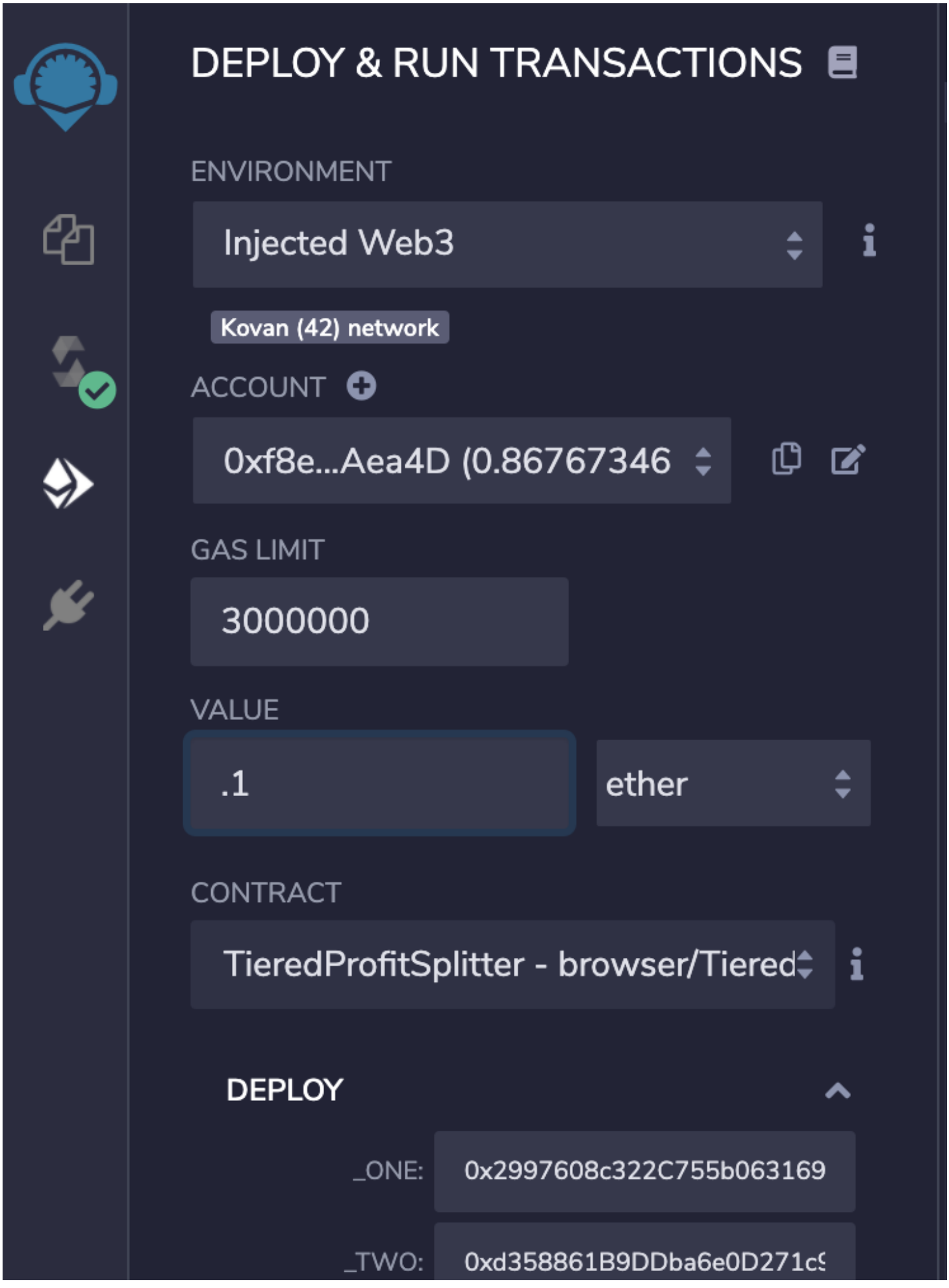
-0.1 ETH

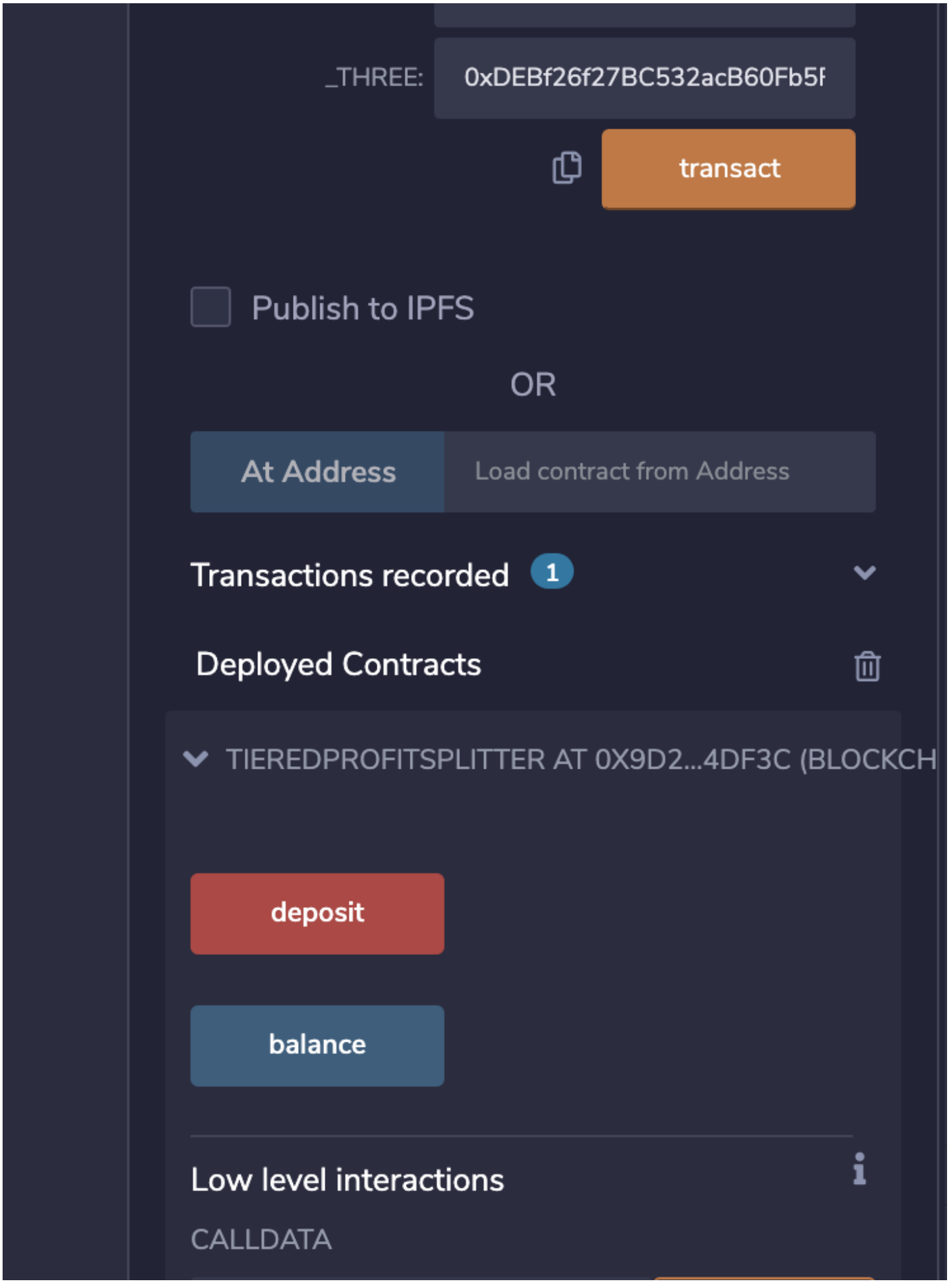


We successfully compile the code, switch the Remix ENVIRONMENT to Injected Web3, enter the three recipient addresses, and "transact" 0 wei to deploy the contract. We Confirm in MetaMask:

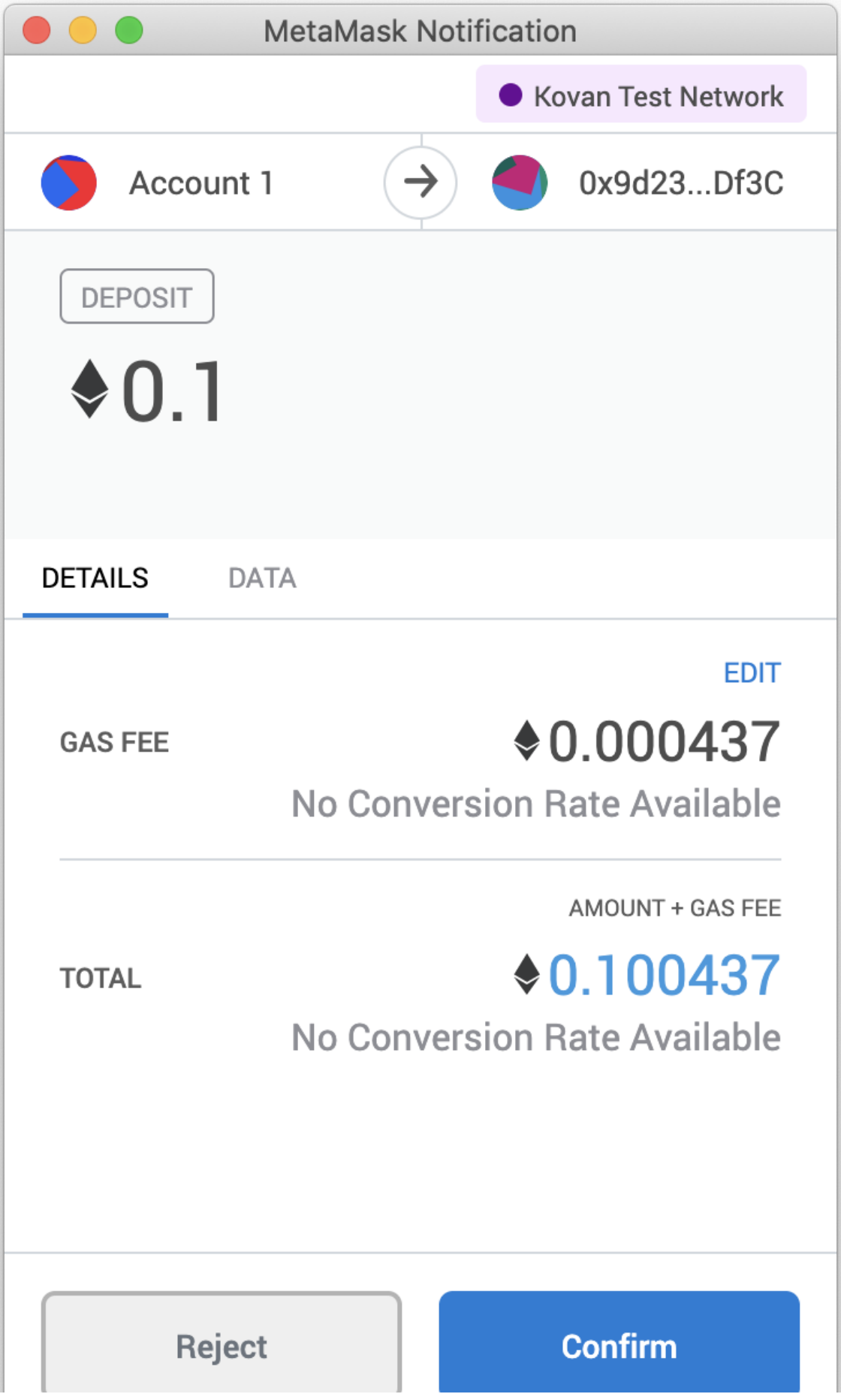

MetaMask Notification
Kovan Test Network
Account 1
New Contract
CONTRACT DEPLOYMENT
0
DETAILS
DATA
EDIT
GAS FEE
0.002337
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
0.002337
No Conversion Rate Available
Reject
Confirm

Now we can see "TIEREDPROFITSPLITTER" in the lower-left dropdown which reveals the "deposit" and "balance" buttons. We choose a VALUE of 0.1 ether to deposit:


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Kovan (42) network
ACCOUNT
0xf8e...Aea4D (0.86767346
GAS LIMIT
3000000
VALUE
.1
ether
CONTRACT
TieredProfitSplitter - browser/Tiered
DEPLOY
_ONE:
0x2997608c322C755b063169
_TWO:
0xd358861B9DDba6e0D271c9


_THREE:
0xDEBf26f27BC532acB60Fb5F
transact
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
1
Deployed Contracts
TIEREDPROFITSPLITTER AT 0X9D2...4DF3C (BLOCKCH
deposit
balance
Low level interactions
CALLDATA

We Confirm in the MetaMask Notification popup:


MetaMask Notification
Kovan Test Network
Account 1
0x9d23...Df3C
DEPOSIT
0.1
DETAILS
DATA
EDIT
GAS FEE
0.000437
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
0.100437
No Conversion Rate Available
Reject
Confirm

We can see the transaction "pending":

Kovan Test Network

Connected

Account 1

0xf8eF...ea4D

0.8677 ETH

BUY **SEND**

Assets | Activity

Queue (1)

Deposit

Pending · remix.ethereum.org

-0.1 ETH

-0.1 ETH

History

We can use MetaMask to view the transaction details and confirm successful execution:

Deposit

Details

View on Etherscan

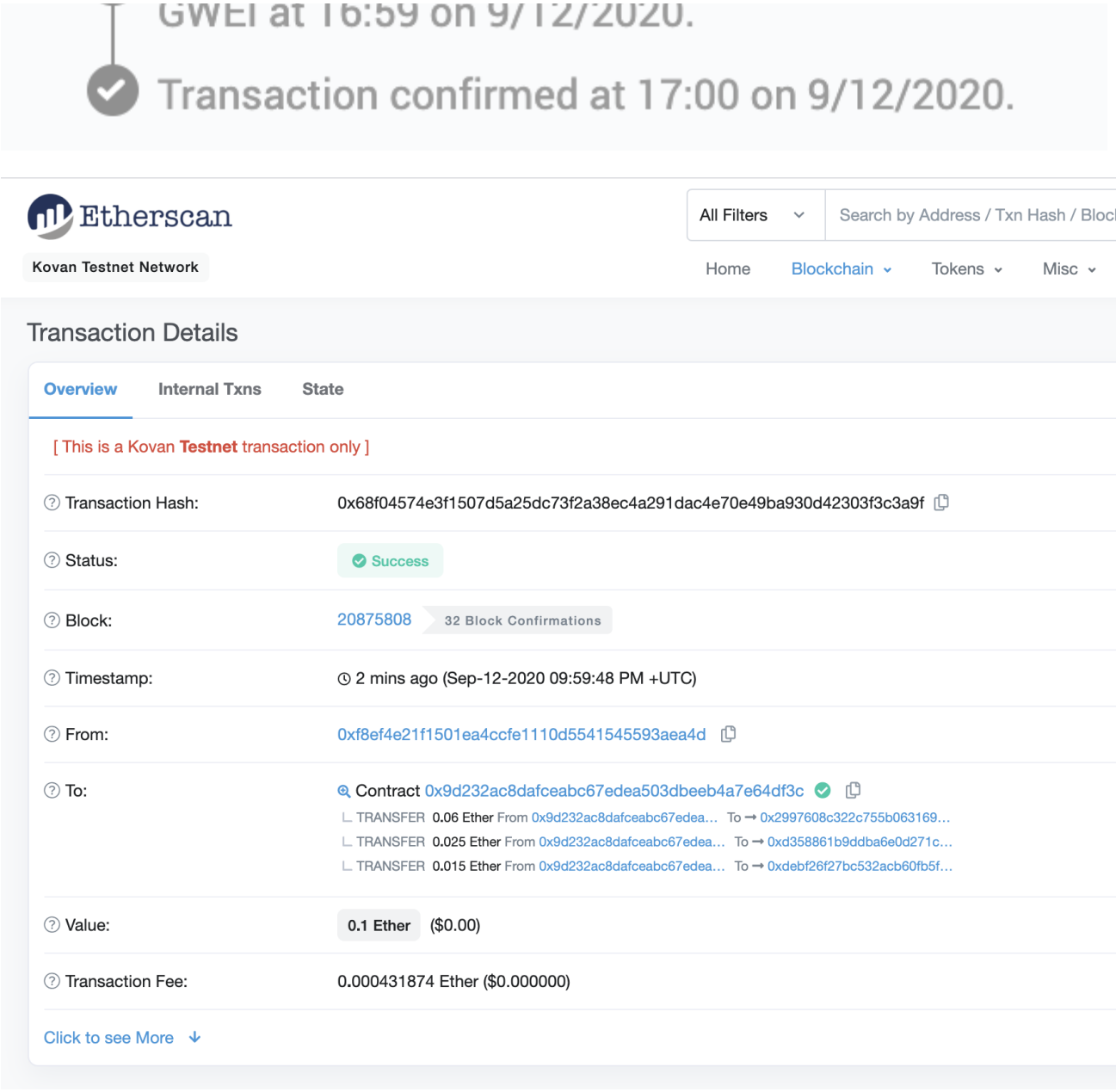
From: 0xf8eF4E21f150... > To: 0x9d232AC8DAFcE...

Transaction

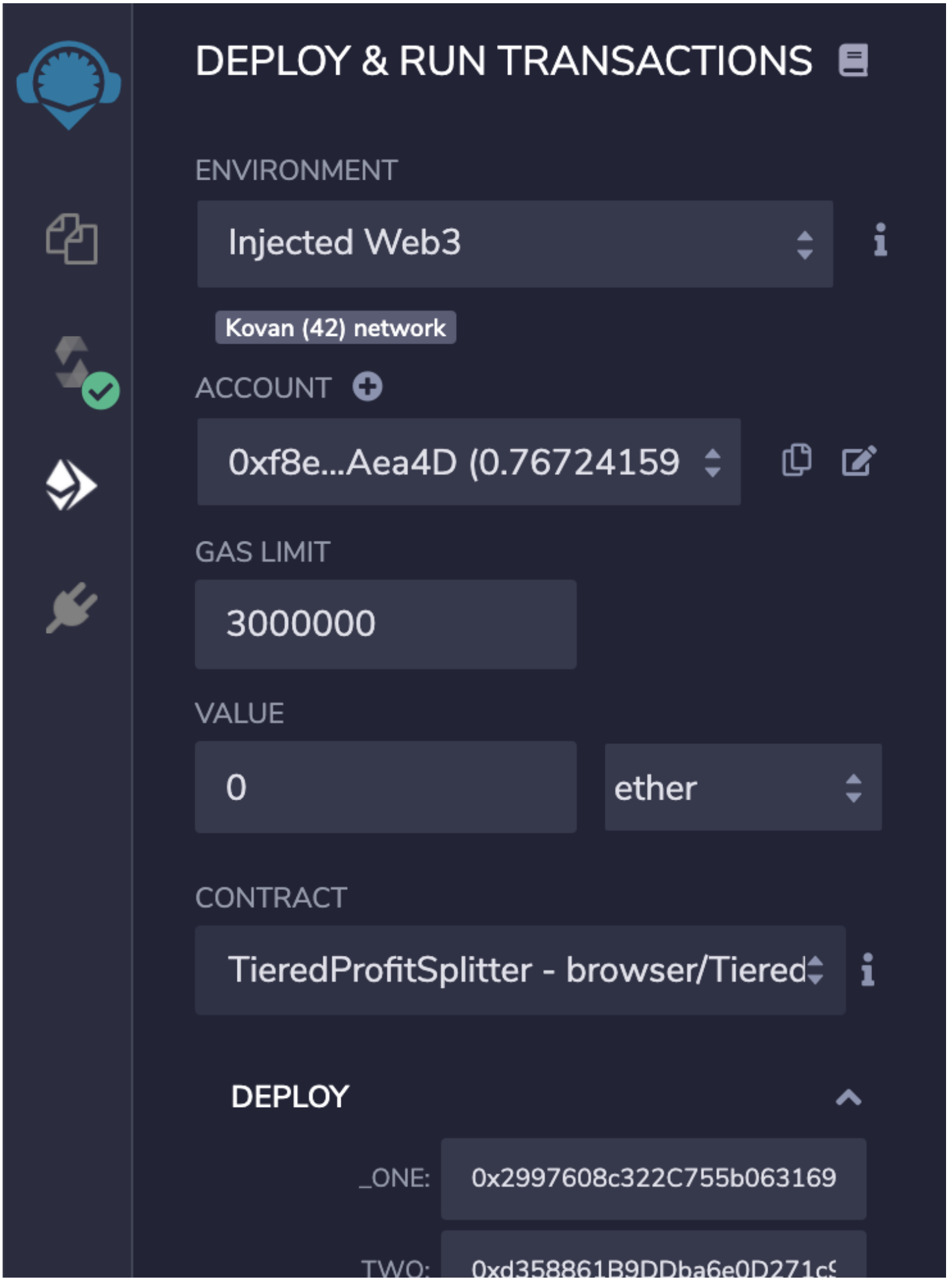
Nonce	11
Amount	0.1 ETH
Gas Limit (Units)	48590
Gas Used (Units)	47986
Gas Price (GWEI)	9
Total	**0.100432 ETH**

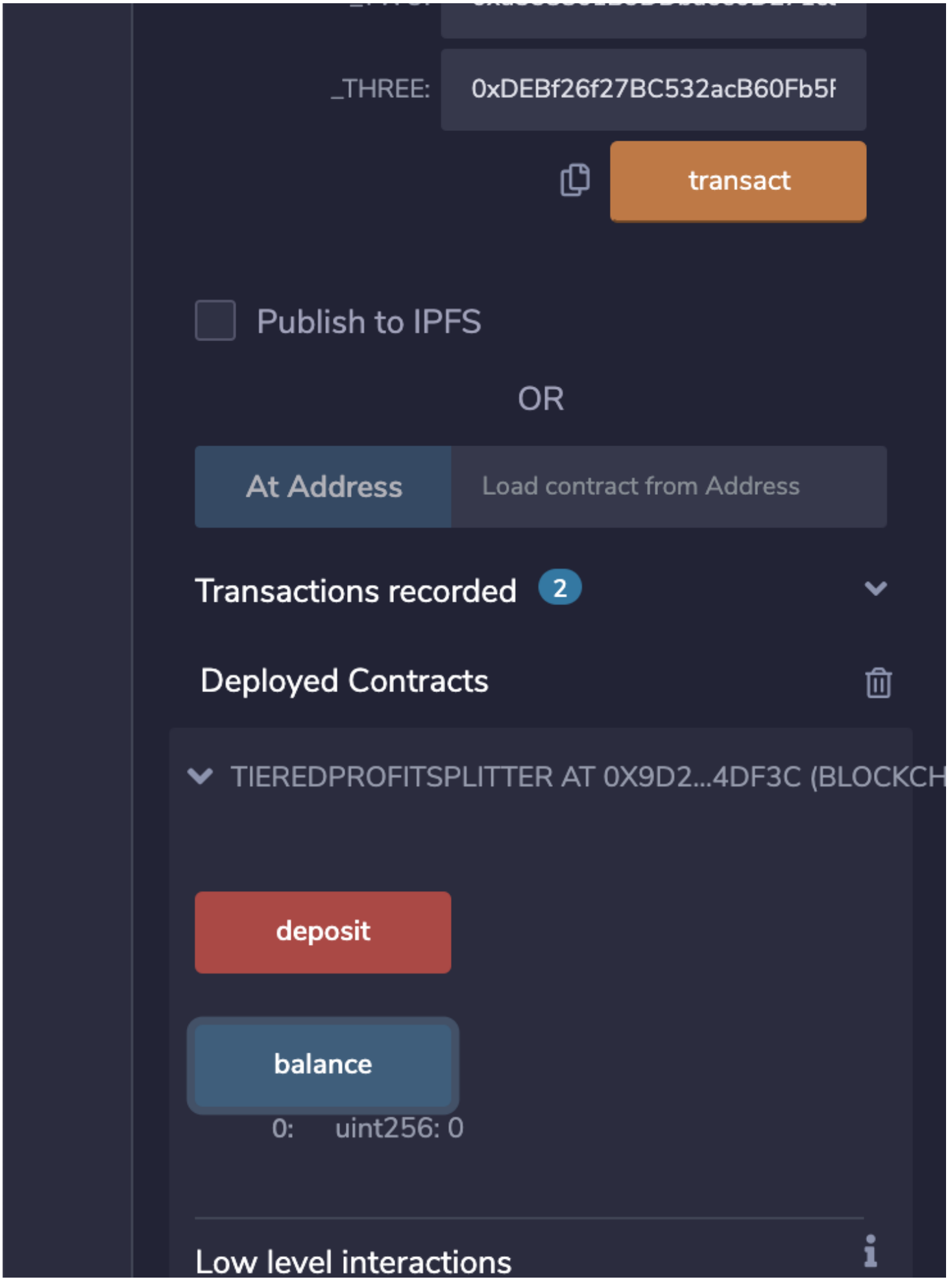
Activity Log

- Transaction created with a value of 0.1 ETH at 16:58 on 9/12/2020.
- Transaction submitted with gas fee of 437310



We now check the "balance" which is 0, as expected:


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Kovan (42) network
ACCOUNT
0xf8e...Aea4D (0.76724159
GAS LIMIT
3000000
VALUE
0
ether
CONTRACT
TieredProfitSplitter - browser/Tiered
DEPLOY
_ONE:
0x2997608c322C755b063169
TWO:
0xd358861B9DDba6e0D271c


_THREE:
0xDEBf26f27BC532acB60Fb5f
transact
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
2
Deployed Contracts
TIEREDPROFITSPLITTER AT 0X9D2...4DF3C (BLOCKCH
deposit
balance
0: uint256: 0
Low level interactions

The testnet address for others to be able to send to is:
0xF40870a74528482E4ba15a8450687b0ee778fe68.

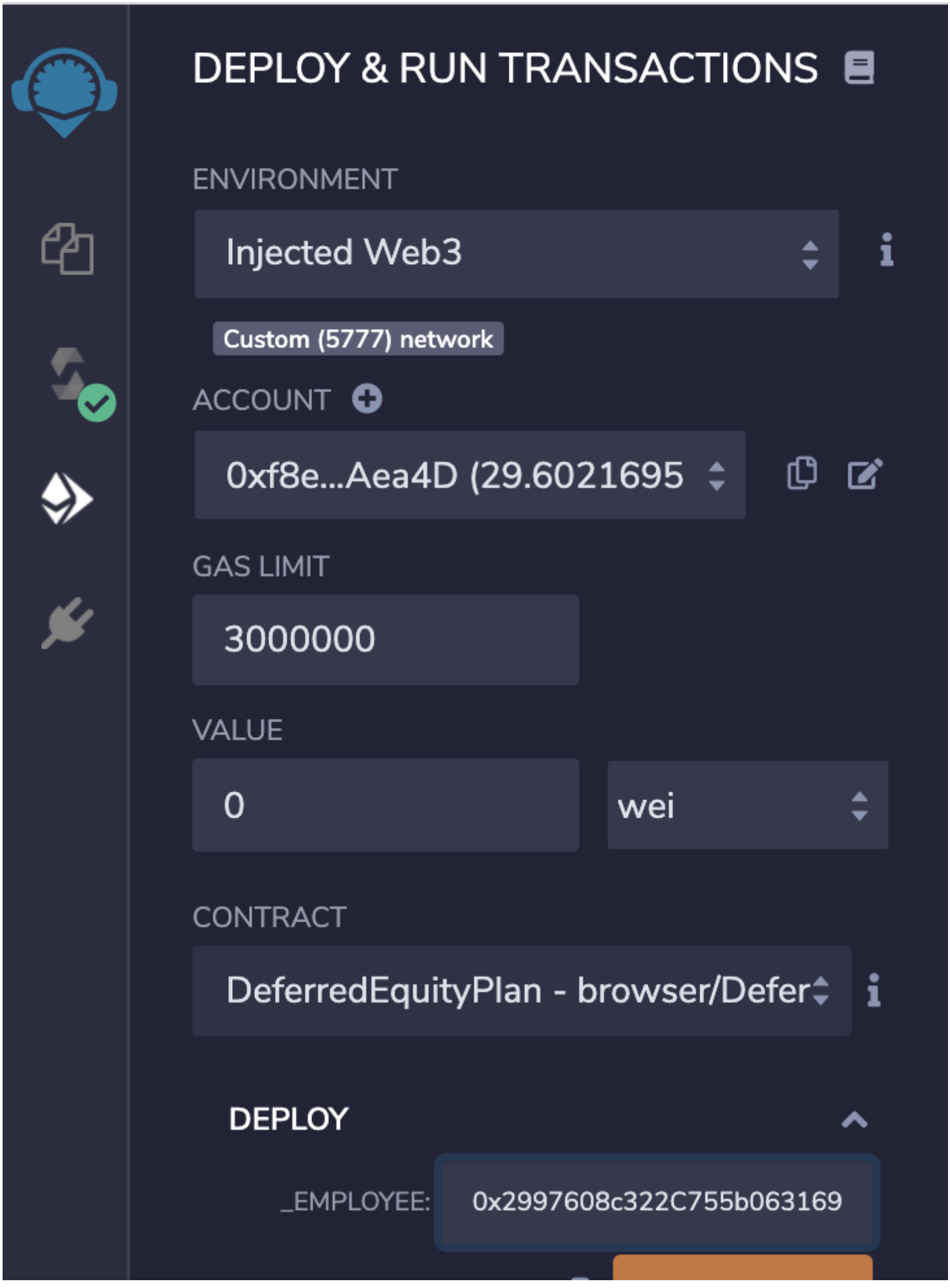
Level Three: The DeferredEquityPlan Contract

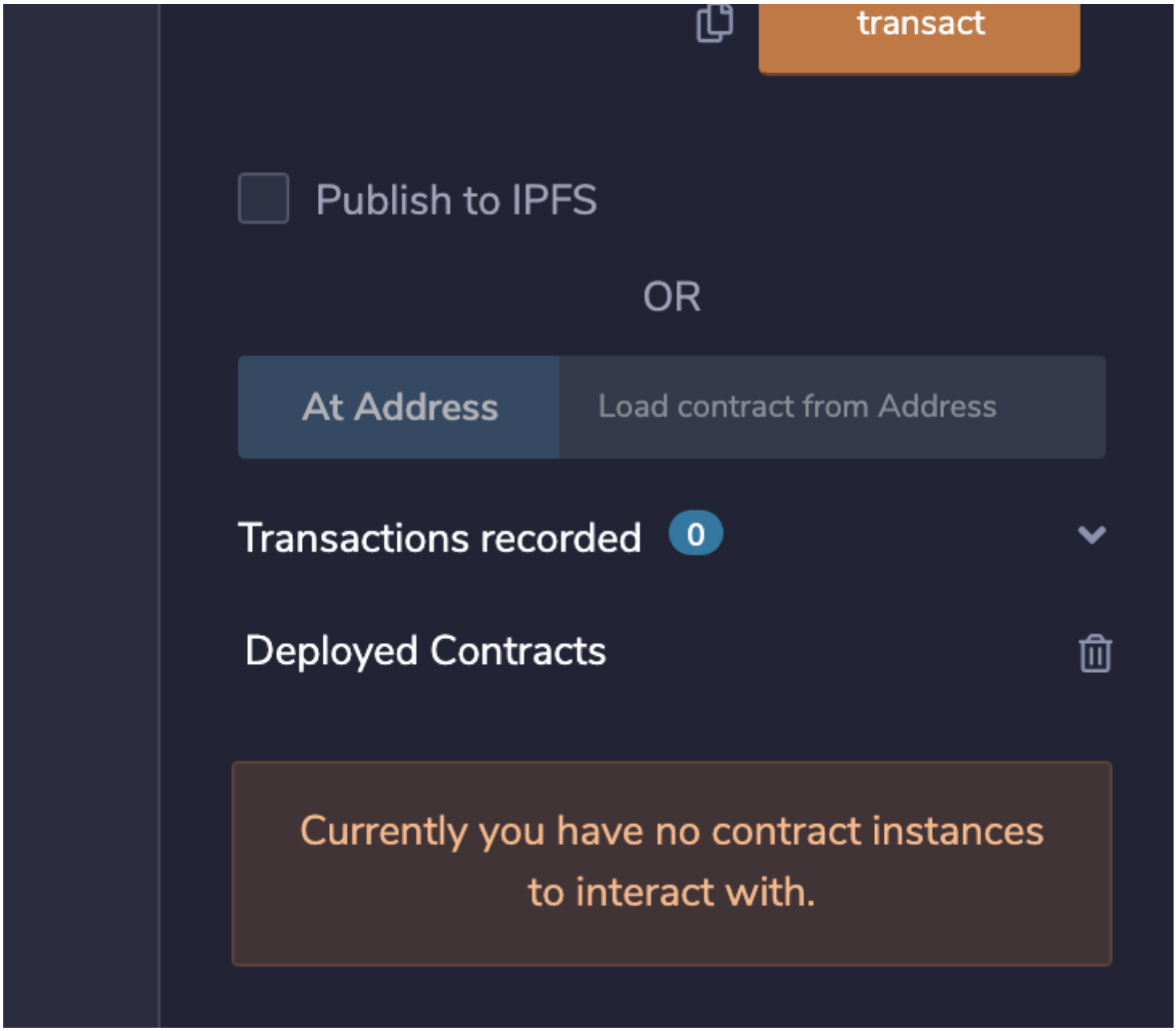
```
pragma solidity ^0.5.0;

// lvl 3: equity plan
contract DeferredEquityPlan {
    uint fakenow = now;
    function fastforward() public {
        fakenow += 100 days;
    }
    address human_resources;

    address payable employee; // bob
    bool active = true; // this employee is active at the start of the contract

    // @TODO: Set the total shares and annual distribution
    uint total_shares = 1000;
    uint annual_distribution = 250;



    uint start_time = fakenow; // permanently store the time this contract was initialized

    // @TODO: Set the `unlock_time` to be 365 days from now
    uint unlock_time = fakenow + 365 days;

    uint public distributed_shares; // starts at 0

    constructor(address payable _employee) public {
        human_resources = msg.sender;
        employee = _employee;
    }

    function distribute() public {
        require(msg.sender == human_resources || msg.sender == employee, "You are not authorized to exe
        require(active == true, "Contract not active.");

        // @TODO: Add "require" statements to enforce that:
        // 1: `unlock_time` is less than or equal to `now`
        require(unlock_time <= fakenow,  "Account is locked!");
        // 2: `distributed_shares` is less than the `total_shares`
        require(distributed_shares < total_shares, "Available shares have been exceeded!");

        // @TODO: Add 365 days to the `unlock_time`
        unlock_time += 365 days;

        // @TODO: Calculate the shares distributed by using the function (now - start_time) / 365 days
        // Make sure to include the parenthesis around (now - start_time) to get accurate results!
        distributed_shares = (fakenow - start_time) / 365 days * annual_distribution;

        // double check in case the employee does not cash out until after 5+ years
        if (distributed_shares > 1000) {
            distributed_shares = 1000;
        }
```

```
    }

    // human_resources and the employee can deactivate this contract at-will
    function deactivate() public {
        require(msg.sender == human_resources || msg.sender == employee, "You are not authorized to dea
        active = false;
    }

    // Since we do not need to handle Ether in this contract, revert any Ether sent to the contract dir
    function() external payable {
        revert("Do not send Ether to this contract!");
    }
}
```

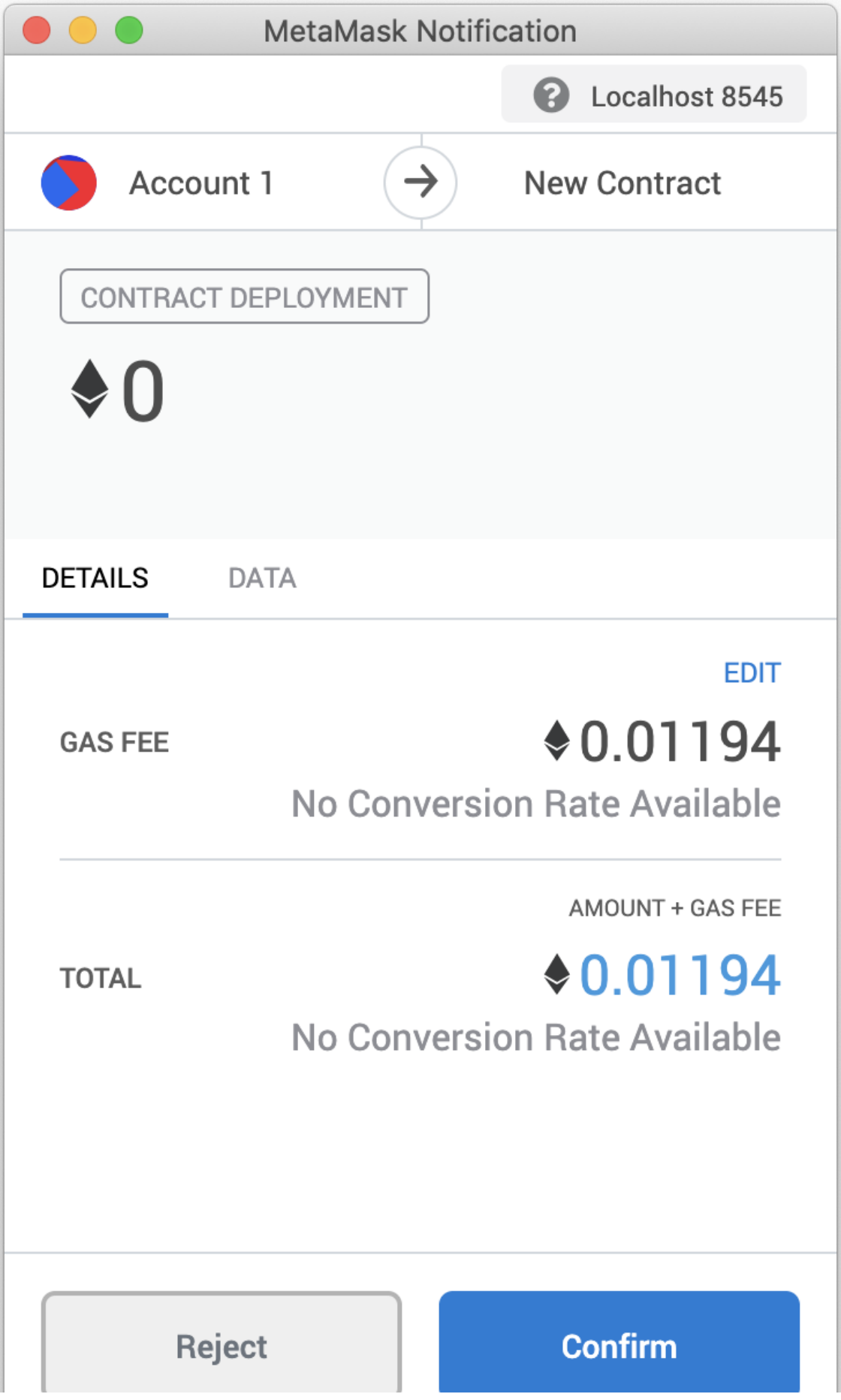
The code works by keeping track of how much time has passed since the start_time, and storing and setting amounts that represent the number of distributed shares the employee owns, enforcing the vetting periods automatically. We use a "fakenow" variable that allows us to skip through time for testing purposes.

We switch back to our Localhost 8545 network and successfully compile the code, then switch the ENVIRONMENT to Injected Web3. We enter the recipient address and click "transact" with a VALUE of 0 wei:

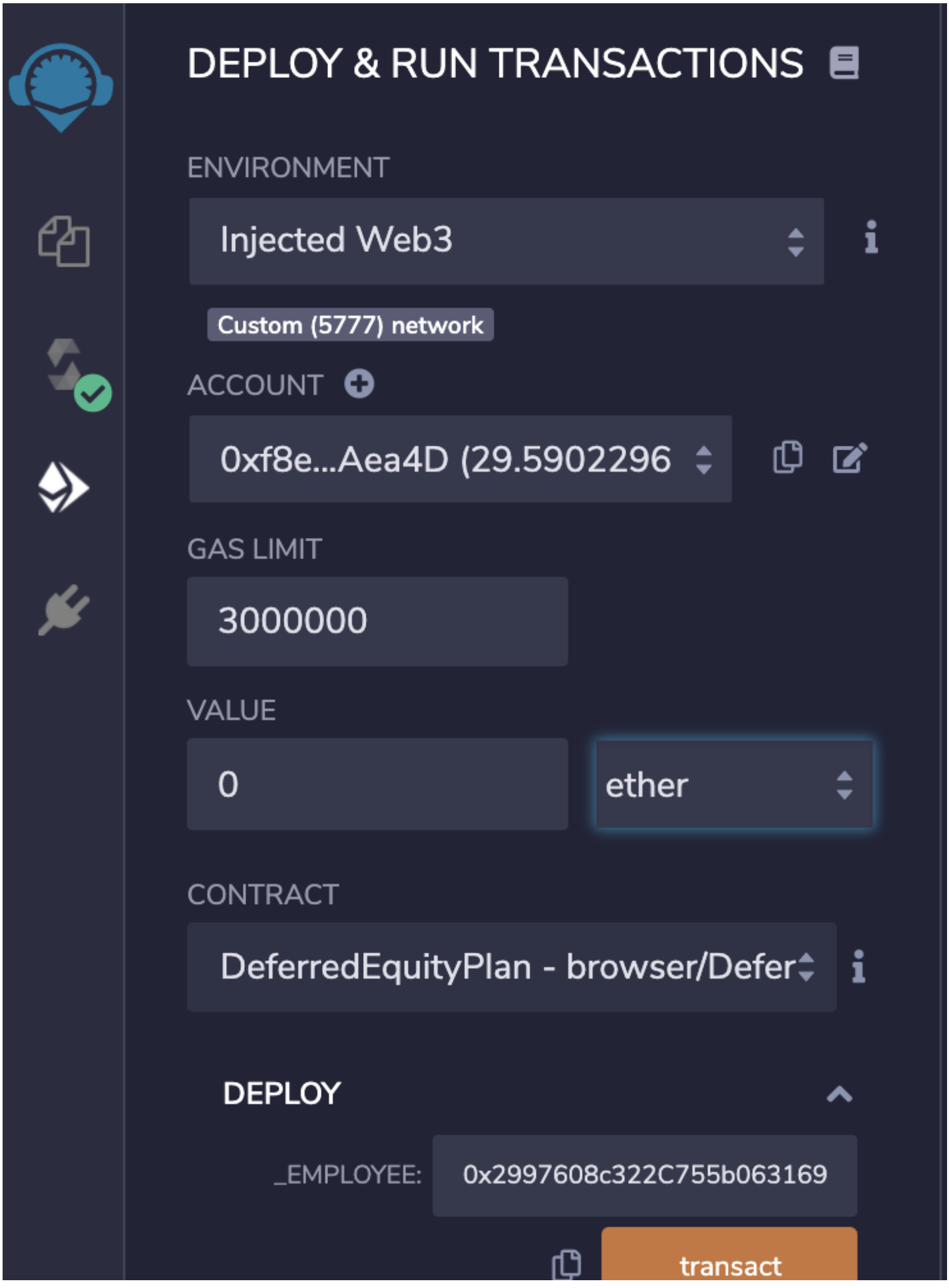

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Custom (5777) network
ACCOUNT
0xf8e...Aea4D (29.6021695
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
DeferredEquityPlan - browser/Defer
DEPLOY
_EMPLOYEE:
0x2997608c322C755b063169

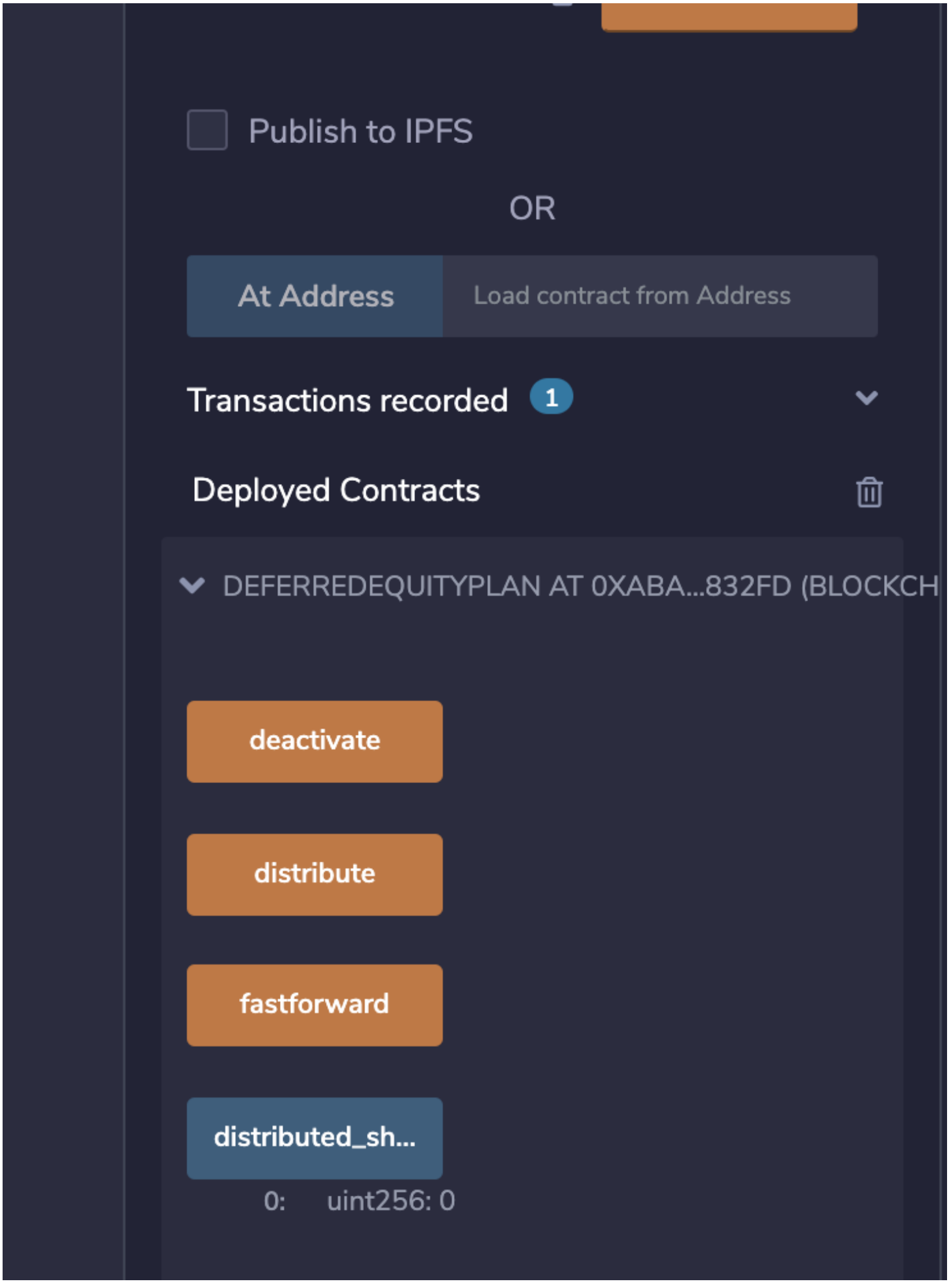


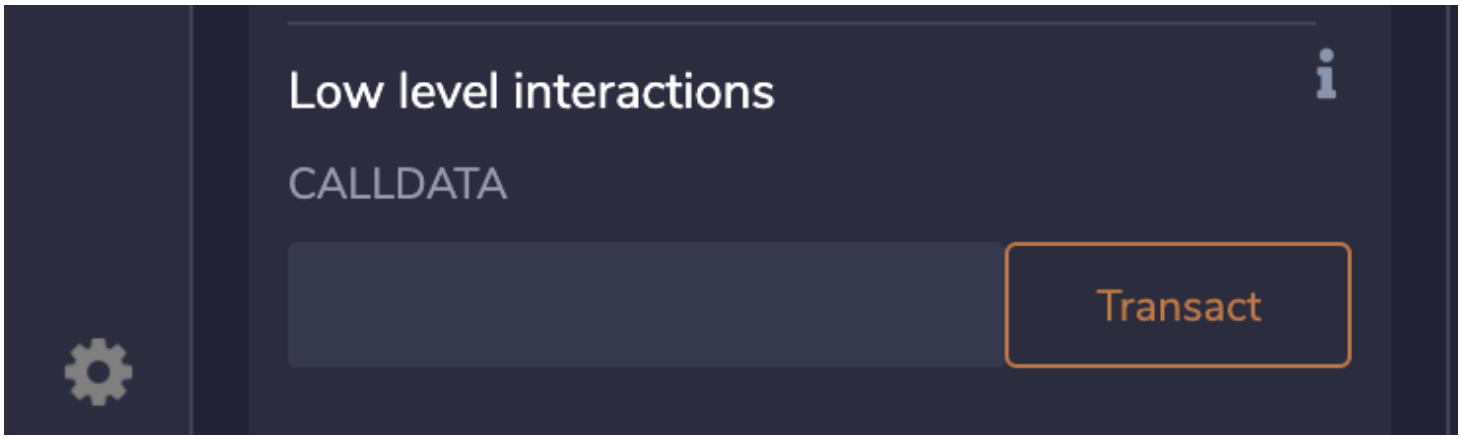
We confirm the deployment in the MetaMask popup:


MetaMask Notification
Localhost 8545
Account 1
New Contract
CONTRACT DEPLOYMENT
0
DETAILS
DATA
EDIT
GAS FEE
0.01194
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
0.01194
No Conversion Rate Available
Reject
Confirm

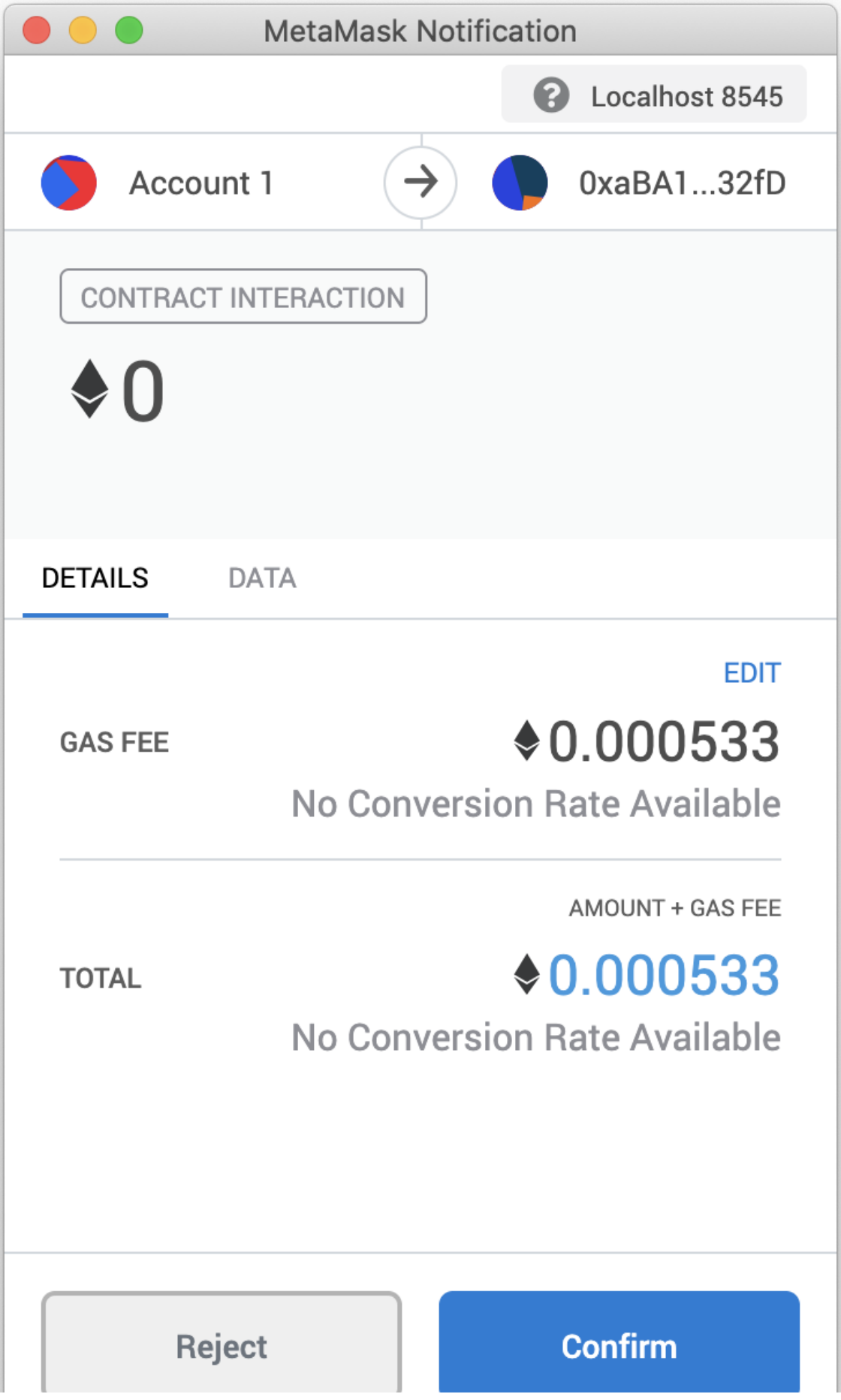
Now we can see the "deactivate," "distribute," "fastforward," and "distributed_shares" buttons in the lower-left dropdown. We choose "distributed_shares" and see that there are 0 distributed shares:


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Custom (5777) network
ACCOUNT
0xf8e...Aea4D (29.5902296
GAS LIMIT
3000000
VALUE
0
ether
CONTRACT
DeferredEquityPlan - browser/Defer
DEPLOY
_EMPLOYEE:
0x2997608c322C755b063169
transact

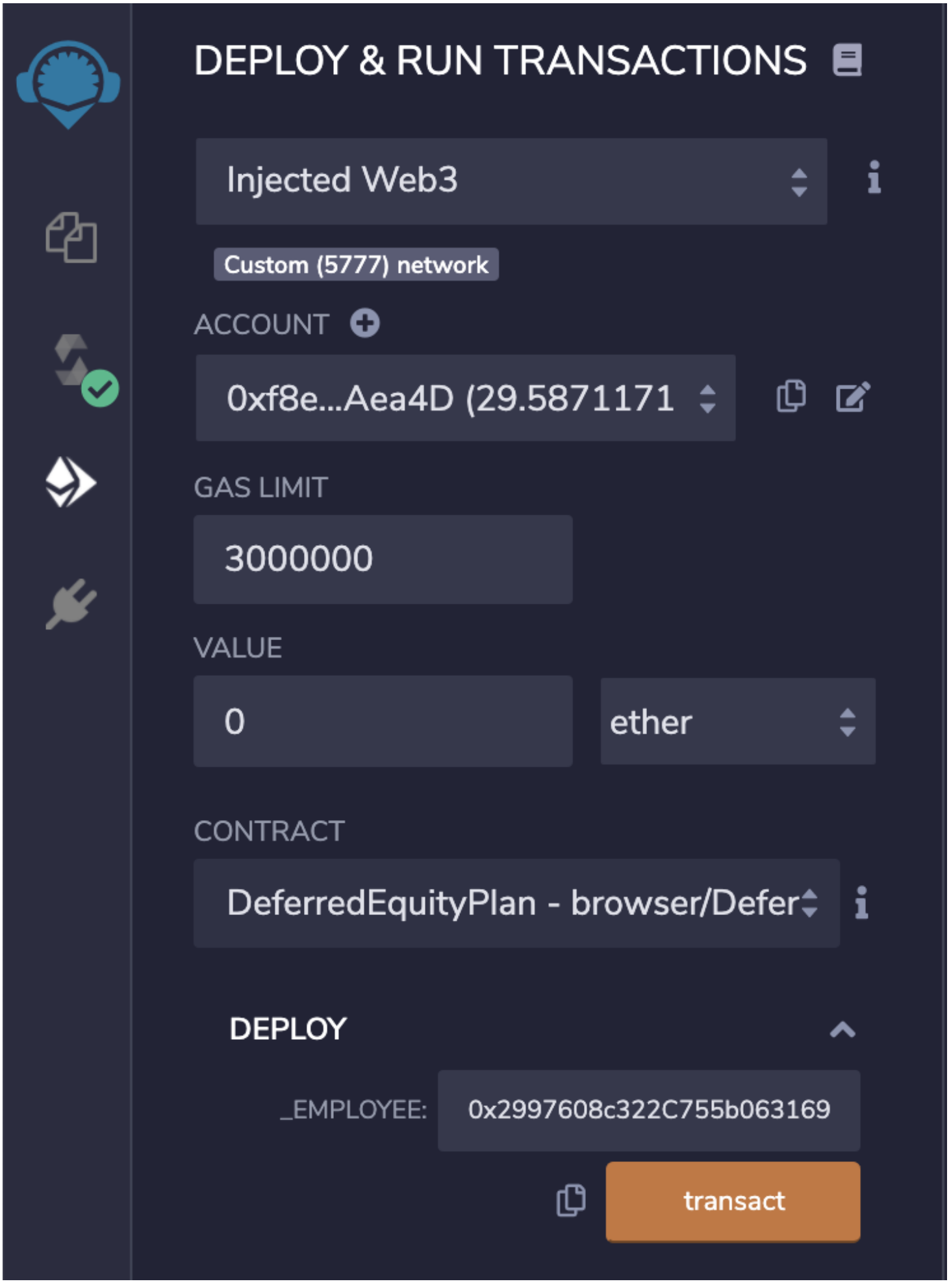

Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 1
Deployed Contracts
DEFERREDEQUITYPLAN AT 0XABA...832FD (BLOCKCH
deactivate
distribute
fastforward
distributed_sh...
0: uint256: 0

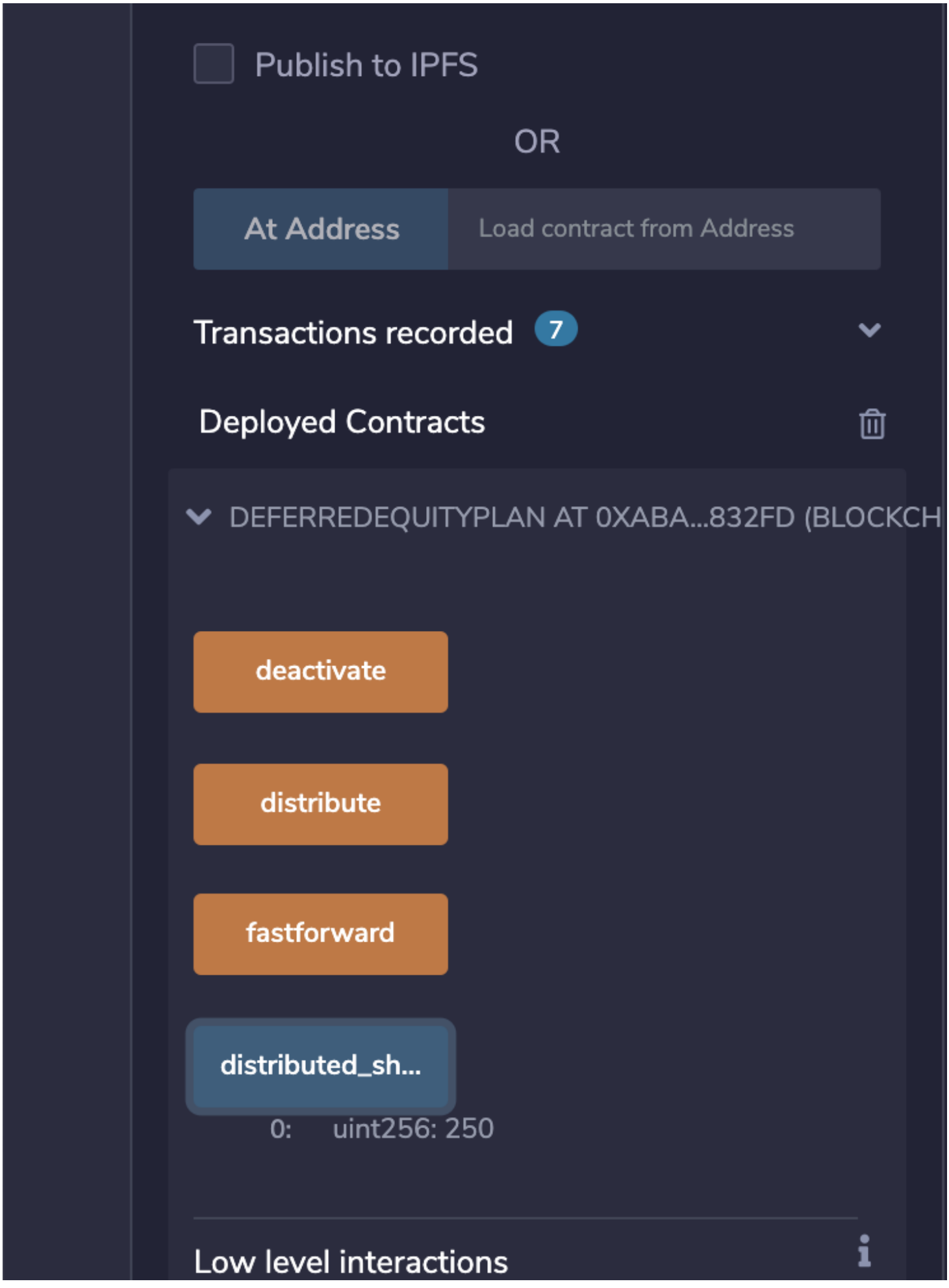


We then choose "fastforward" and Confirm in the MetaMask popup:

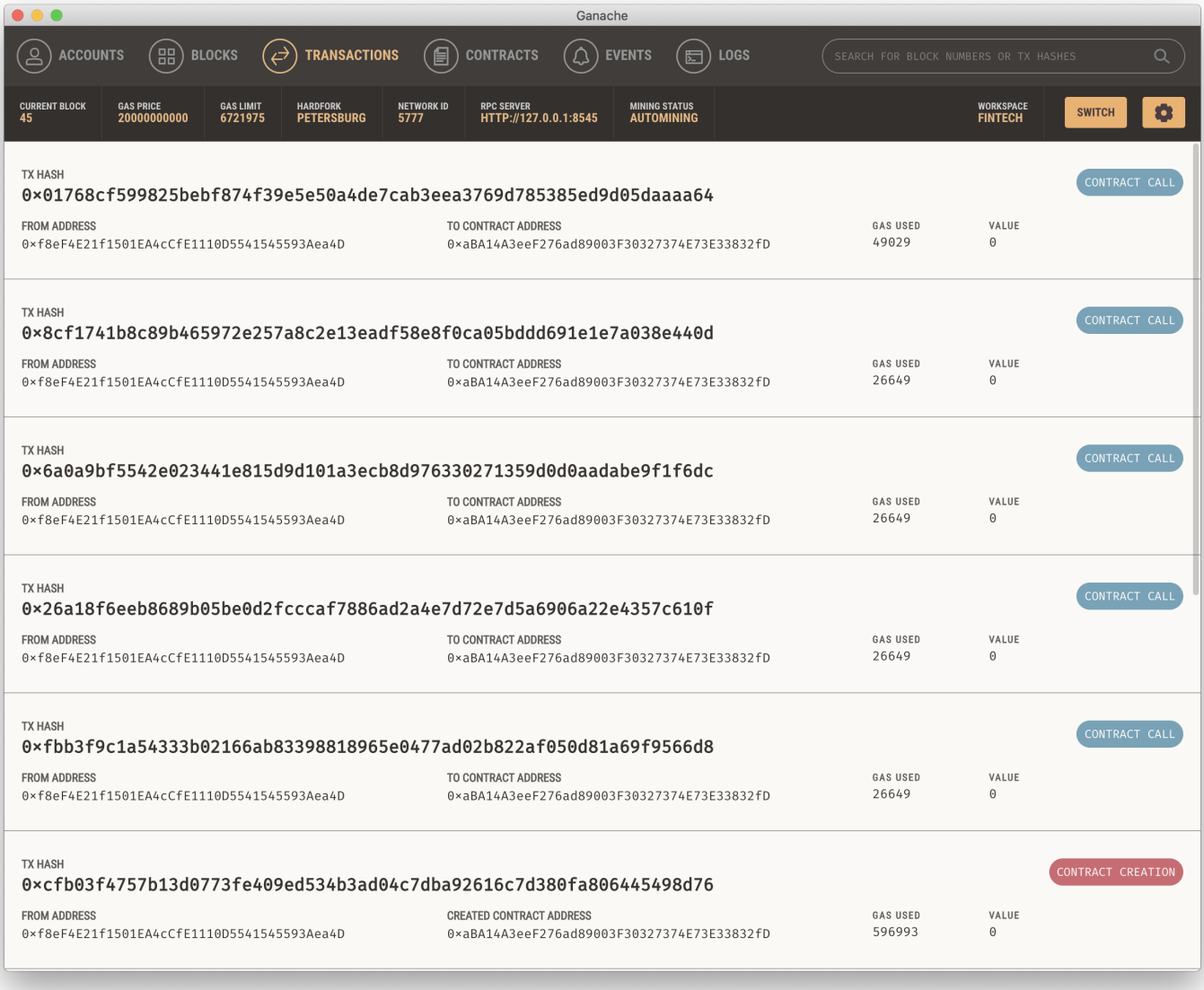

MetaMask Notification
Localhost 8545
Account 1
0xaBA1...32fD
CONTRACT INTERACTION
0
DETAILS
DATA
EDIT
GAS FEE
0.000533
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
0.000533
No Conversion Rate Available
Reject
Confirm

We still see 0 distributed shares because not enough time has passed, so we click "fastforward" three more times, confirming in the MetaMask popup each time. Now we are able to choose "distribute" and Confirm in the MetaMask popup. Now when we click "distributed_shares" we see there are 250 distributed shares, as expected!

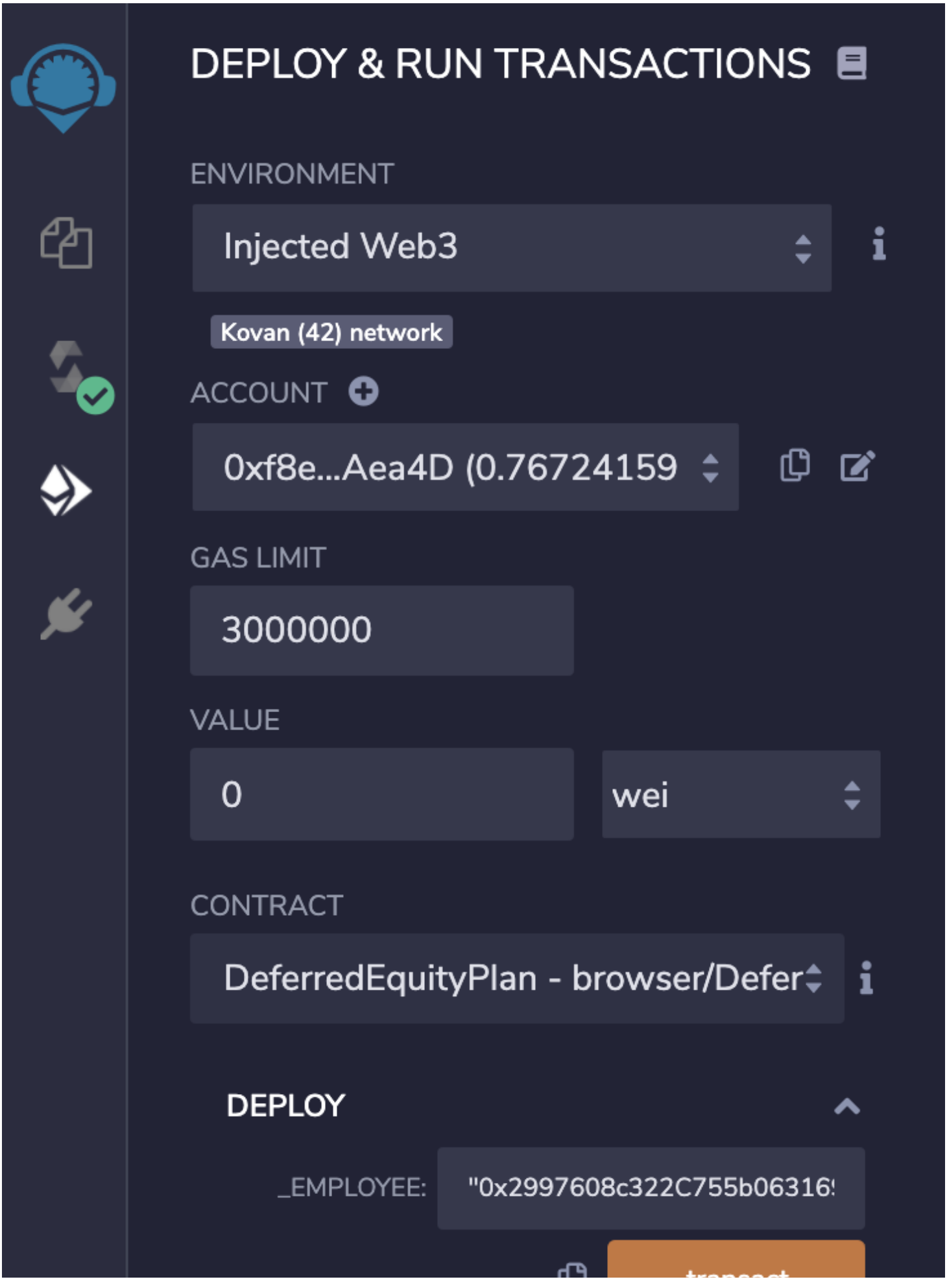

DEPLOY & RUN TRANSACTIONS
Injected Web3
Custom (5777) network
ACCOUNT
0xf8e...Aea4D (29.5871171
GAS LIMIT
3000000
VALUE
0
ether
CONTRACT
DeferredEquityPlan - browser/Defer
DEPLOY
_EMPLOYEE:
0x2997608c322C755b063169
transact

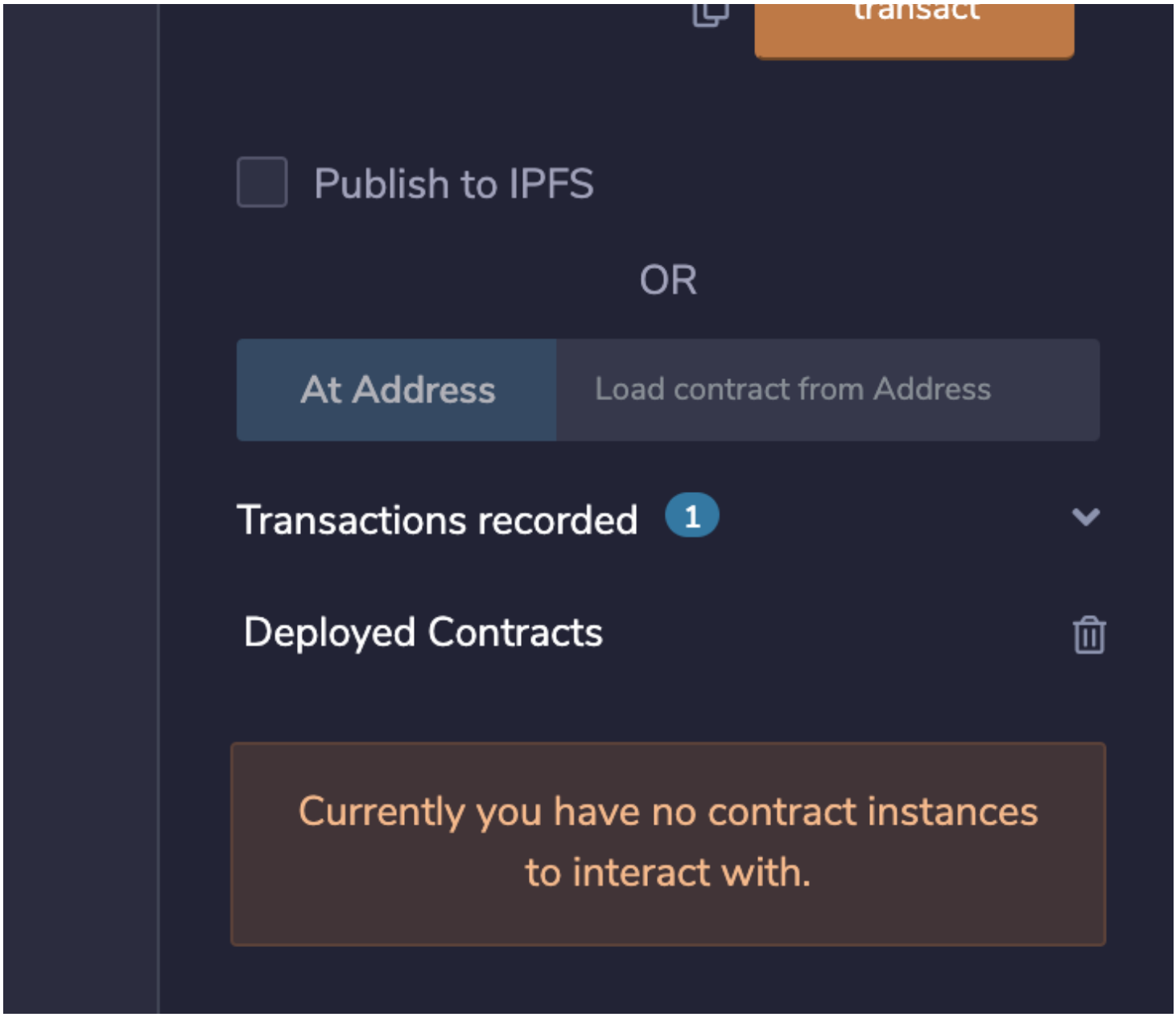

Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 7
Deployed Contracts
DEFERREDEQUITYPLAN AT 0XABA...832FD (BLOCKCH
deactivate
distribute
fastforward
distributed_sh...
0: uint256: 250
Low level interactions

We can see these recent transactions in Ganache:

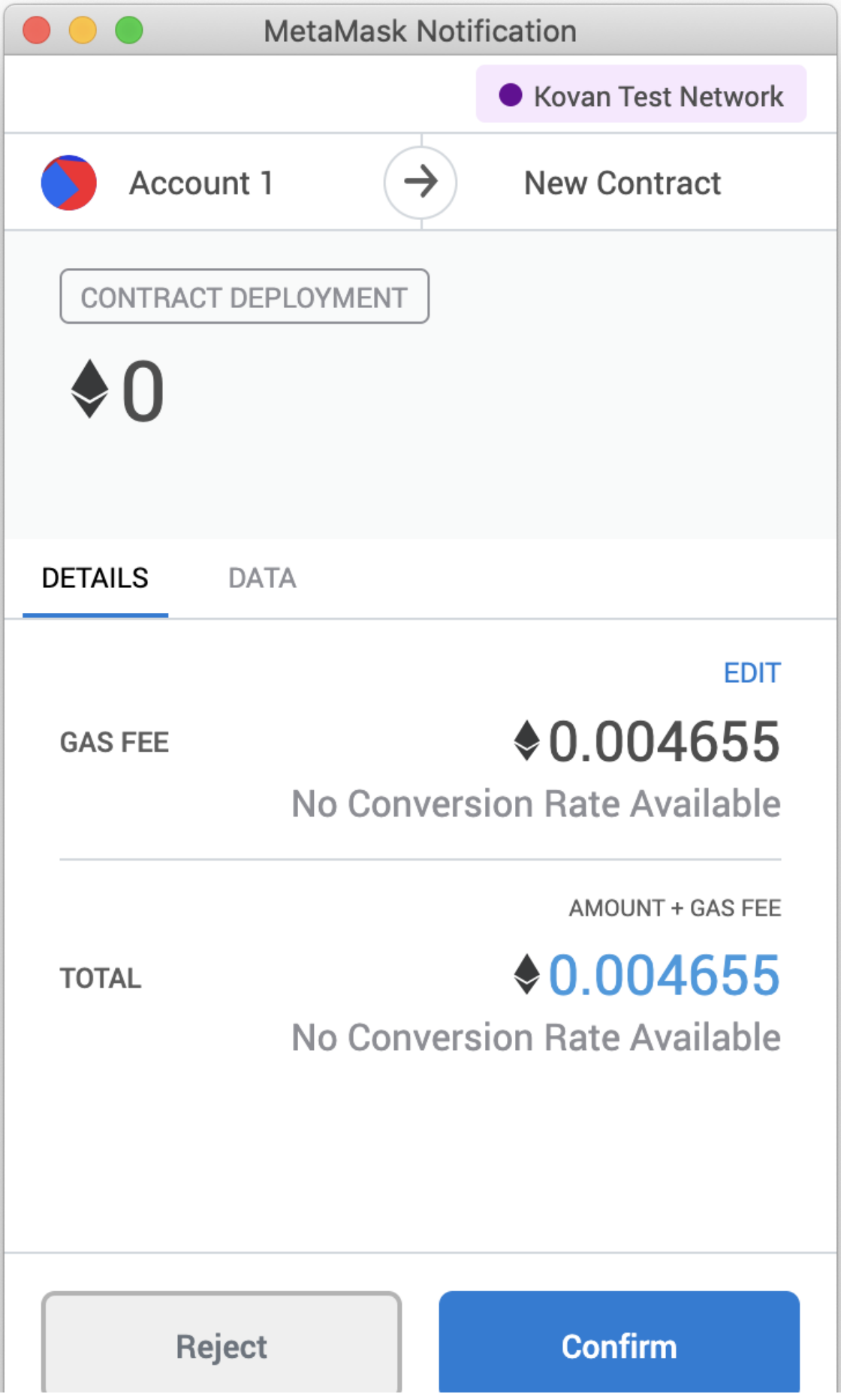


Now we switch back to the Kovan Test Network in MetaMask, successfully compile the code, switch the Remix ENVIRONMENT to Injected Web3, enter the recipient address, and "transact" 0 wei to deploy the contract:

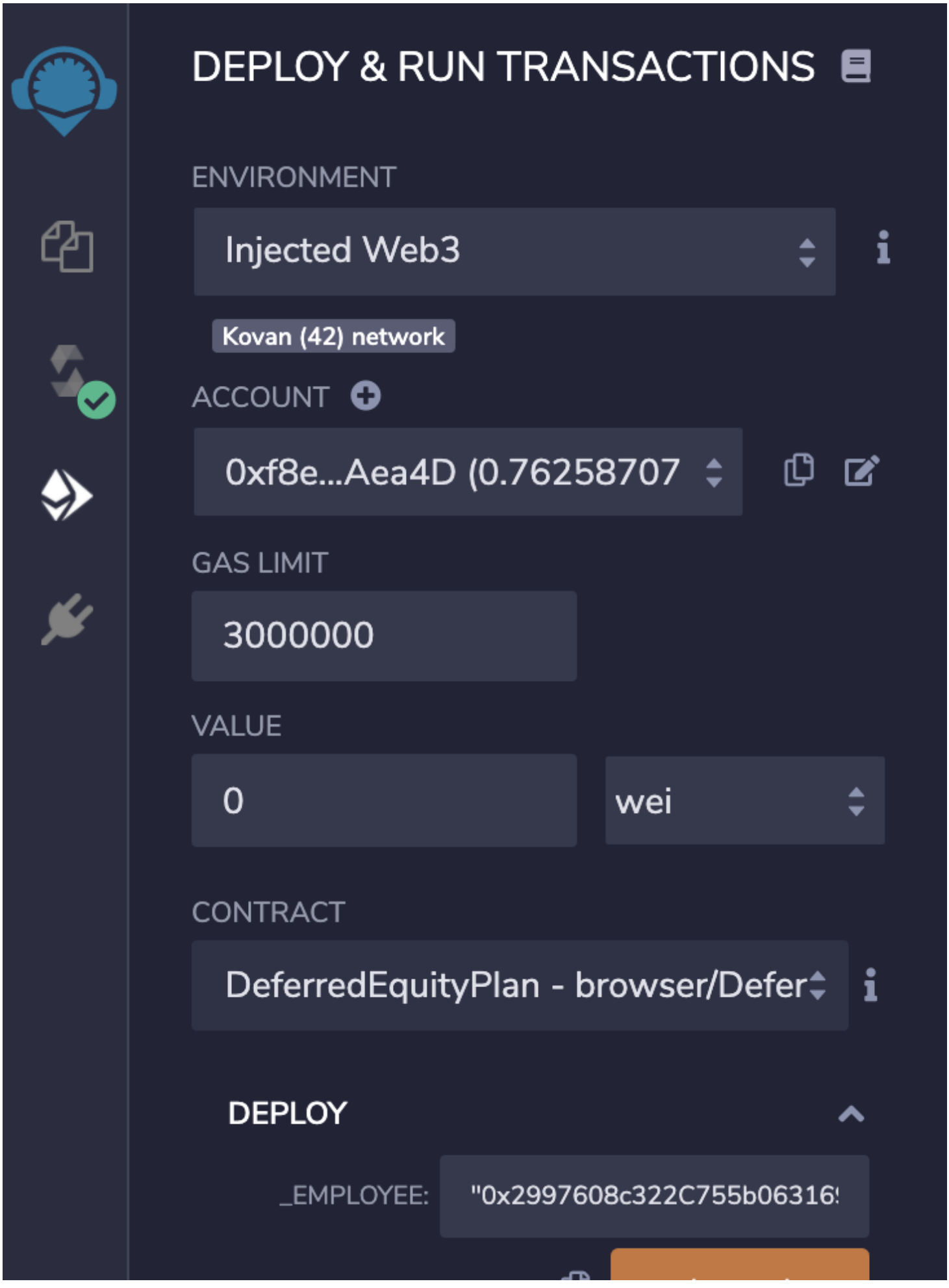

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Kovan (42) network
ACCOUNT
0xf8e...Aea4D (0.76724159
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
DeferredEquityPlan - browser/Defer
DEPLOY
_EMPLOYEE:
"0x2997608c322C755b06316

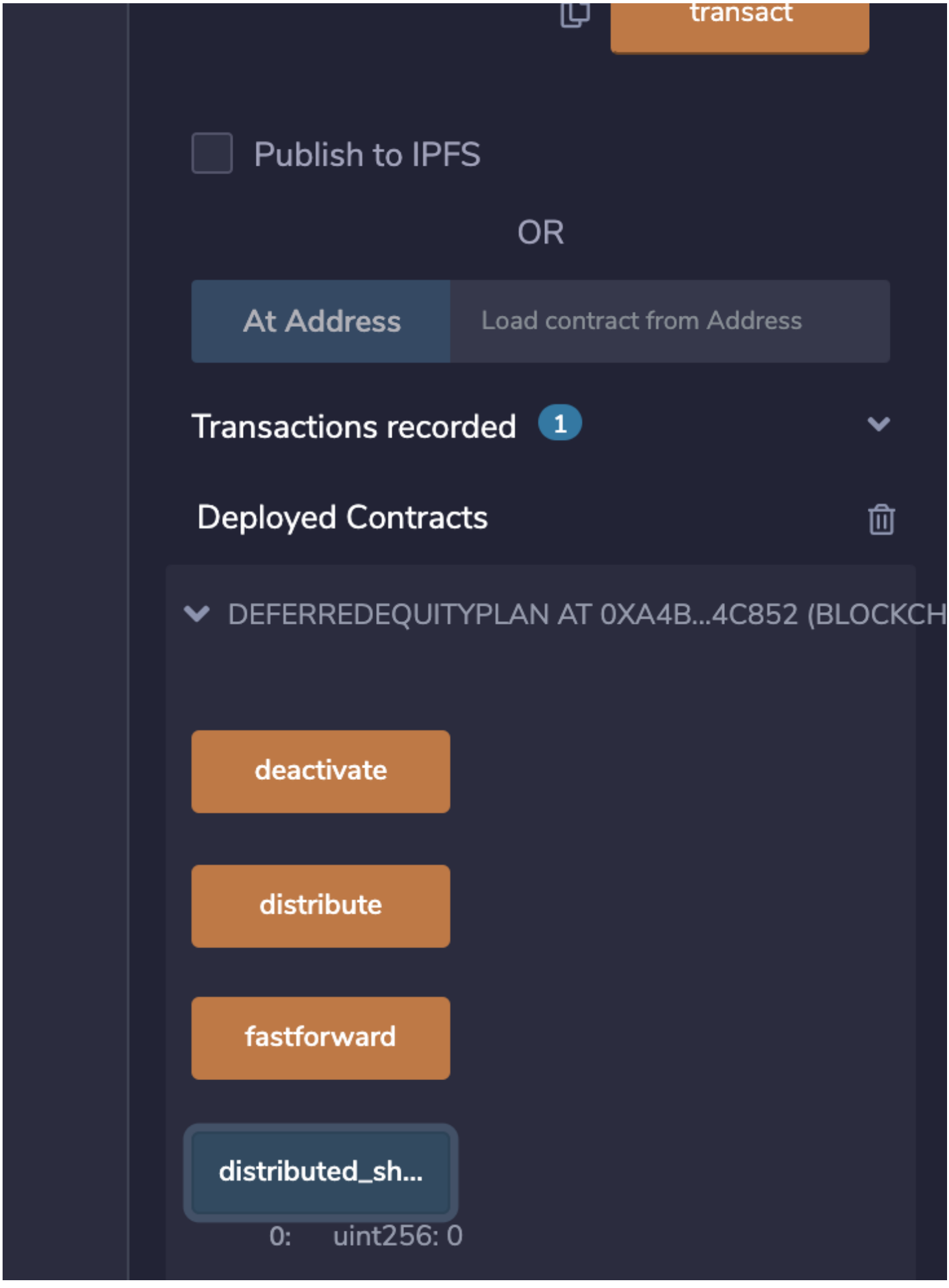


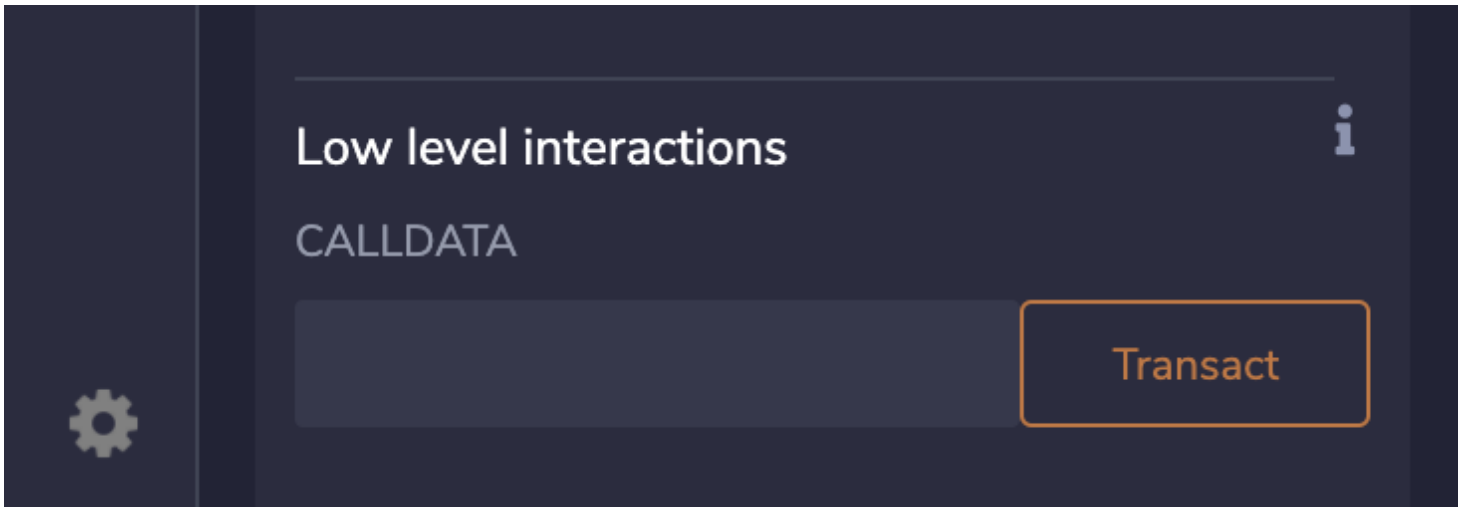
We Confirm in MetaMask:


MetaMask Notification
Kovan Test Network
Account 1
New Contract
CONTRACT DEPLOYMENT
0
DETAILS
DATA
EDIT
GAS FEE
0.004655
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
0.004655
No Conversion Rate Available
Reject
Confirm

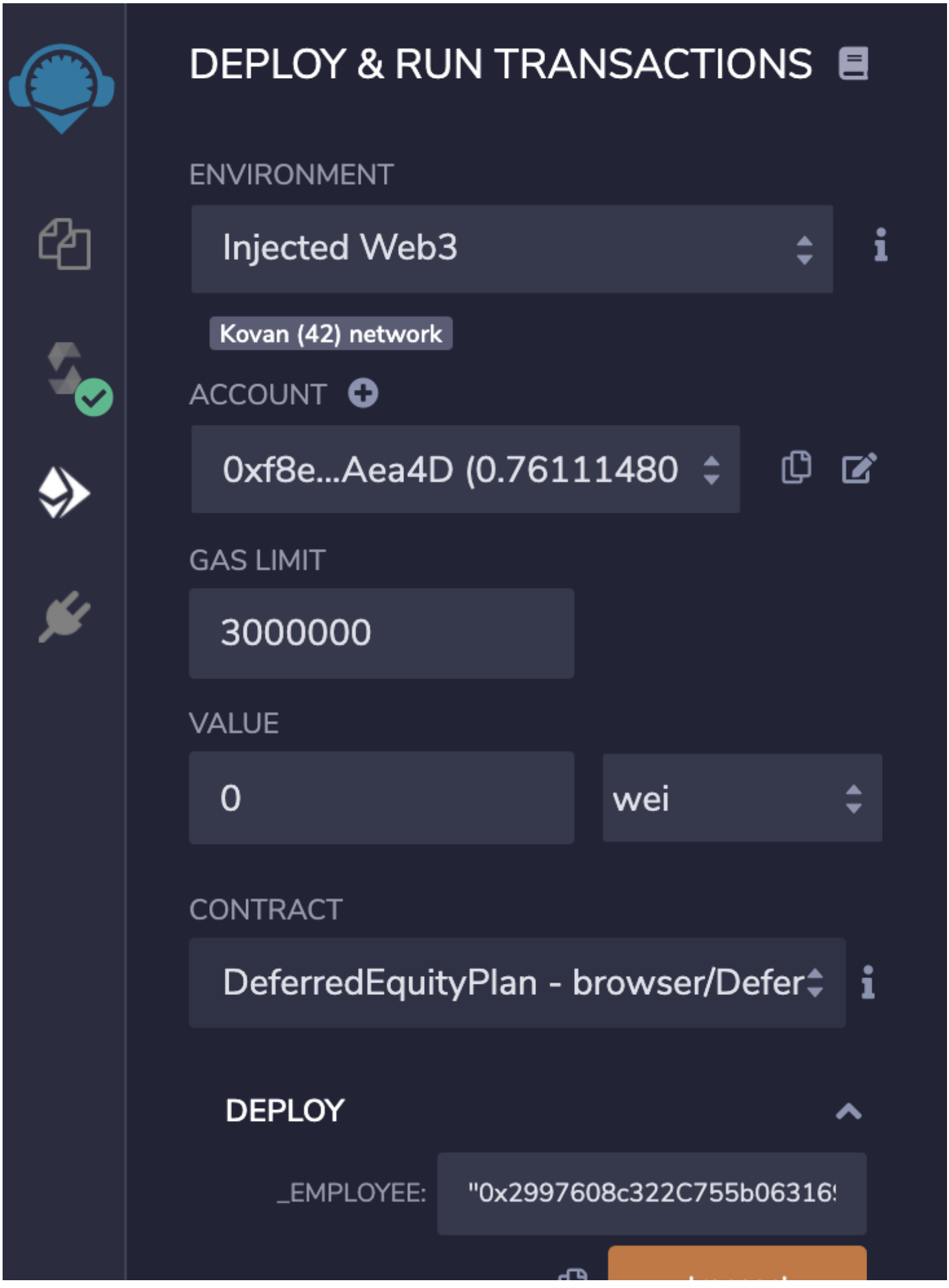
Now we can see the "deactivate," "distribute," "fastforward," and "distributed_shares" buttons in the lower-left dropdown. We choose "distributed_shares" and see that there are 0 distributed shares:

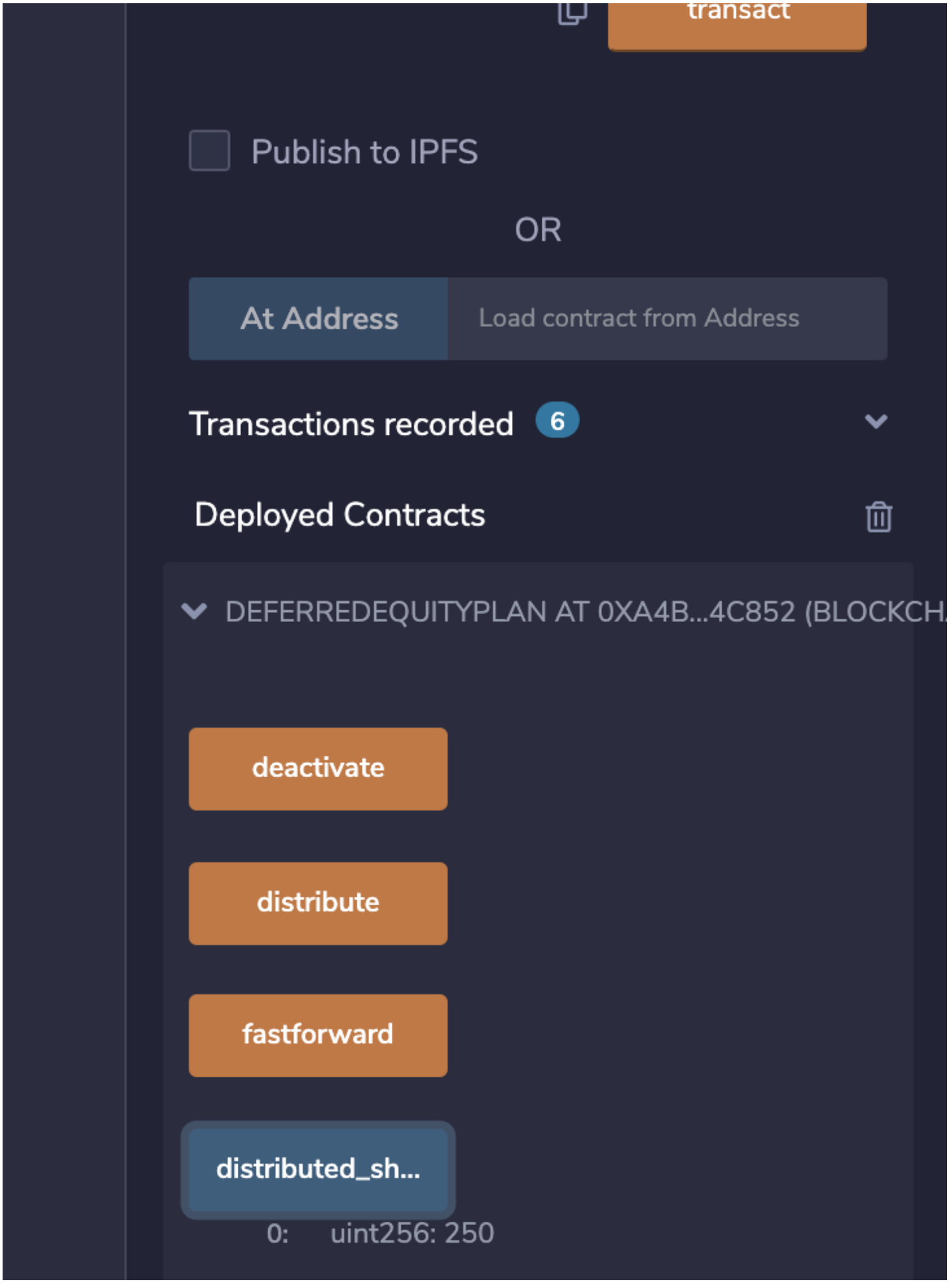

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Kovan (42) network
ACCOUNT
0xf8e...Aea4D (0.76258707
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
DeferredEquityPlan - browser/Defer
DEPLOY
_EMPLOYEE:
"0x2997608c322C755b06316

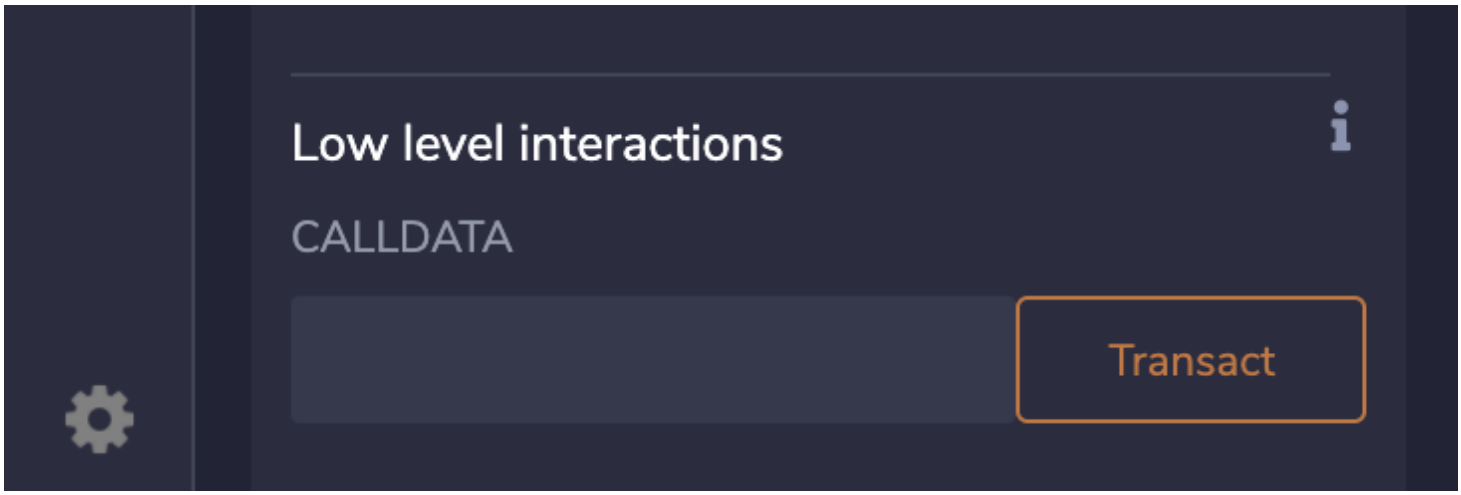

transact
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 1
Deployed Contracts
DEFERREDEQUITYPLAN AT 0XA4B...4C852 (BLOCKCH
deactivate
distribute
fastforward
distributed_sh...
0: uint256: 0



We then choose "fastforward" four times, Confirming in the MetaMask popup. We click "distribute" and now when we click "distributed_shares" we see there are 250 distributed shares, as expected!


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Kovan (42) network
ACCOUNT
0xf8e...Aea4D (0.76111480
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
DeferredEquityPlan - browser/Defer
DEPLOY
_EMPLOYEE:
"0x2997608c322C755b06316


transact
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 6
Deployed Contracts
DEFERREDEQUITYPLAN AT 0XA4B...4C852 (BLOCKCH
deactivate
distribute
fastforward
distributed_sh...
0: uint256: 250



The testnet address for others to be able to send to is: 0xF40870a74528482E4ba15a8450687b0ee778fe68.