COLUMBIA | ENGINEERING
The Fu Foundation School of Engineering and Applied Science

PROJECT THREE

SMART CONTRACTS FOR CONSTRUCTION

Developed By
Crystal Evans, Patrick Ruejoma, Jeff Smith, Mike Cutno, Russell Moore, Sheng Gao

COUNTER PARTY RISK PREVENTION

SMART CONTRACTS FOR CONSTRUCTION

PROJECT OVERVIEW

Our company will house third-party escrow agents who work closely with our in-house attorneys and relationship management team to establish clear communication and facilitate efficient escrow transactions between developers and contractors to avoid Class action/litigation settlements.

We work with issuers and obligors within the construction industry to serve as trustees and paying agents for various transactions.

Paying agent
Collateral agent
Settlement agent
Escrow agent
Custodian
Depositary

COUNTER PARTY RISK PREVENTION

Many of the key assumptions on which our third party DAPPS – Escrow Financial model is related to the construction stage of the projects, legal review and helps address:

1. the need to identify and limit the liabilities during the construction stage, beyond the construction costs;
2. the requirement for a fixed start date and end date for the project's commercial operations
3. the need to confirm the Developers capacity to pay the Contractors commercial operation expenses per phase of the project; and
4. Validate that the actual construction deliverables meets the specifications initially agreed upon.

WHAT IS COUNTERPARTY RISK?

Counterparty risk is the likelihood or probability that one of those involved in a transaction might default on its contractual obligation.

.

WHAT IS COUNTERPARTY RISK?

Counterparty risk is also known as default risk. Default risk is the chance that companies or individuals will be unable to make the required payments on their debt obligations.

.

WHAT IS COUNTERPARTY RISK?

Counterparty risk can exist in credit, investment, and trading transactions.

.

COUNTER PARTY RISK PREVENTION

CONSTRUCTION INDUSTRY TRENDS

INDUSTRY TRENDS

As one of the largest industries in the US (and the world), the construction market remains lucrative.

Beyond the coronavirus-induced hiccups that hit the industry in early 2020, the US construction industry is constantly changing and adjusting to new economic climates around the globe.

Construction statistics from 2020 show that there is not only a lot of growth in the market — but also increased risks.

INDUSTRY TRENDS

Today, construction data is an invaluable asset for contractors that can assist them in their decision-making — from preconstruction to collecting payment.

Payment processes in the construction industry is a focal point of research in recent years.

As published in the 2020 National Construction Payment Report conducted in Quarter 1 of 2020, a survey of over 540 construction professionals found that 80% of construction businesses spend a significant portion of their work week seeking payments for their work.

INDUSTRY TRENDS

Additionally, just 50% of those businesses said they received payment within 30 days of invoicing.

US construction spending totaled $1.3 trillion in 2019.

INDUSTRY TRENDS

Credit and payments in construction

63.2% of contractors say they “sometimes” get paid on time according to terms in their contract.

Speech bubble illustration saying "49% of construction payments were not paid on time in 2019.”
26.9% of contractors say it takes between 31-45 days to receive payment after invoicing.

The median DSO in the construction/engineering industry was 90 days in 2019.

INDUSTRY TRENDS

49% of construction payments were not paid on time in 2019.

Mechanics liens: In a 2019 survey of 503 contractors, 74% of contractors claimed they filed a mechanics lien within the last year.

INDUSTRY TRENDS

Construction dispute statistics

North America's average length of construction disputes was 17.6 months and average value of construction disputes was $18.8 million in 2019.

The same study cites mediation as the most common method of construction alternative
dispute resolution that year.

The most common construction dispute cause was listed as contractor/subcontractor "failing to understand and/or comply" with a contractual obligation in 2019.

COUNTER PARTY RISK PREVENTION

SMART CONTRACTS

HOW SMART CONTRACTS CAN AVOID LITIGATION

Our team of professionals will use immutable contract clauses to secure deals between all parties and enable each party to upload and easily share supporting documents to validate requirements, deliverables, and milestones (Licenses & Permits, Safety Trainings Certificates, W9 Forms, Construction Insurance Certificates, Construction Bonds - 1.5M), send and receive payment per validation.

How Smart Contracts Can Avoid Litigation

Our company will help negotiate and review construction contracts to make sure the all terms and payments are clear to all parties.

CONSTRUCTION CONTRACT AGREEMENT

PARTIES

- This Construction Contract Agreement (hereinafter referred to as the **"Agreement"**) is entered into on _____ (the **"Effective Date"**), by and between _____, with an address of _____ (hereinafter referred to as the **"Constructor"**), and _____, with an address of _____ (hereinafter referred to as the **"Client"**) (collectively referred to as the **"Parties"**).

CONSTRUCTION PROPERTY

- The Property that is to be constructed is located at the following address:

_____

_____

SCOPE OF WORK

The Constructor agrees to perform the construction described below:

1. _____
2. _____
3. _____
4. _____
5. _____
6. _____
7. _____
8. _____
9. _____
10. _____

- In addition to the aforementioned, the Constructor agrees to abide by and perform all the work that is shown on the construction plan available on the property's site.

MINGW64:/c/Users/Russell

```
Russell@DESKTOP-JL9AU1F MINGW64 ~
$ gpg --gen-key
gpg (GnuPG) 2.2.29-unknown; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Note: Use "gpg --full-generate-key" for a full featured key generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name: Russell Moore
Email address: rdm5181@gmail.com
You selected this USER-ID:
    "Russell Moore <rdm5181@gmail.com>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit? o
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
```

MINGW64:/c/Users/Russell/Documents/Fintech-Workspace/Python_Project/FinalProject

```
Russell@DESKTOP-JL9AU1F MINGW64 ~
$ gpg --list-keys
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   2  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 2u
gpg: next trustdb check due at 2024-11-16
/c/Users/Russell/.gnupg/pubring.kbx
---------------------------------
pub   rsa3072 2022-11-17 [SC] [expires: 2024-11-16]
      2413642B93F0686018F3FA9CCD156617EF665914
uid           [ultimate] Russell Moore <rdm5181@gmail.com>
sub   rsa3072 2022-11-17 [E] [expires: 2024-11-16]

pub   rsa3072 2022-11-23 [SC] [expires: 2024-11-22]
      A0252FF010B1A5CD702E4B760089AB73F59E40E7
uid           [ultimate] Russell Moore <rdm5181@gmail.com>
sub   rsa3072 2022-11-23 [E] [expires: 2024-11-22]


Russell@DESKTOP-JL9AU1F MINGW64 ~
$ cd Documents

Russell@DESKTOP-JL9AU1F MINGW64 ~/Documents
$ cd Fintech-Workspace

Russell@DESKTOP-JL9AU1F MINGW64 ~/Documents/Fintech-Workspace
$ cd Python_Project

Russell@DESKTOP-JL9AU1F MINGW64 ~/Documents/Fintech-Workspace/Python_Project
$ cd FinalProject

Russell@DESKTOP-JL9AU1F MINGW64 ~/Documents/Fintech-Workspace/Python_Project/FinalProject (main)
$ gpg --encrypt --recipient "Russell Moore" construction_contract_agreement.jpg

Russell@DESKTOP-JL9AU1F MINGW64 ~/Documents/Fintech-Workspace/Python_Project/FinalProject (main)
$
```

IPFS

QmHash/bafyHash

Browse

STATUS

FILES

EXPLORE

PEERS

SETTINGS

Files

37KiB
FILES

61MiB
ALL BLOCKS

+ Import

Name ↑	Pin Status	Size
construction_contract_agreement.jpg.gpg Qme1FR6KGtRLVzhY1QXuwybPx17NcW8UK6ZZ4DJxTTdJNq		37 KiB

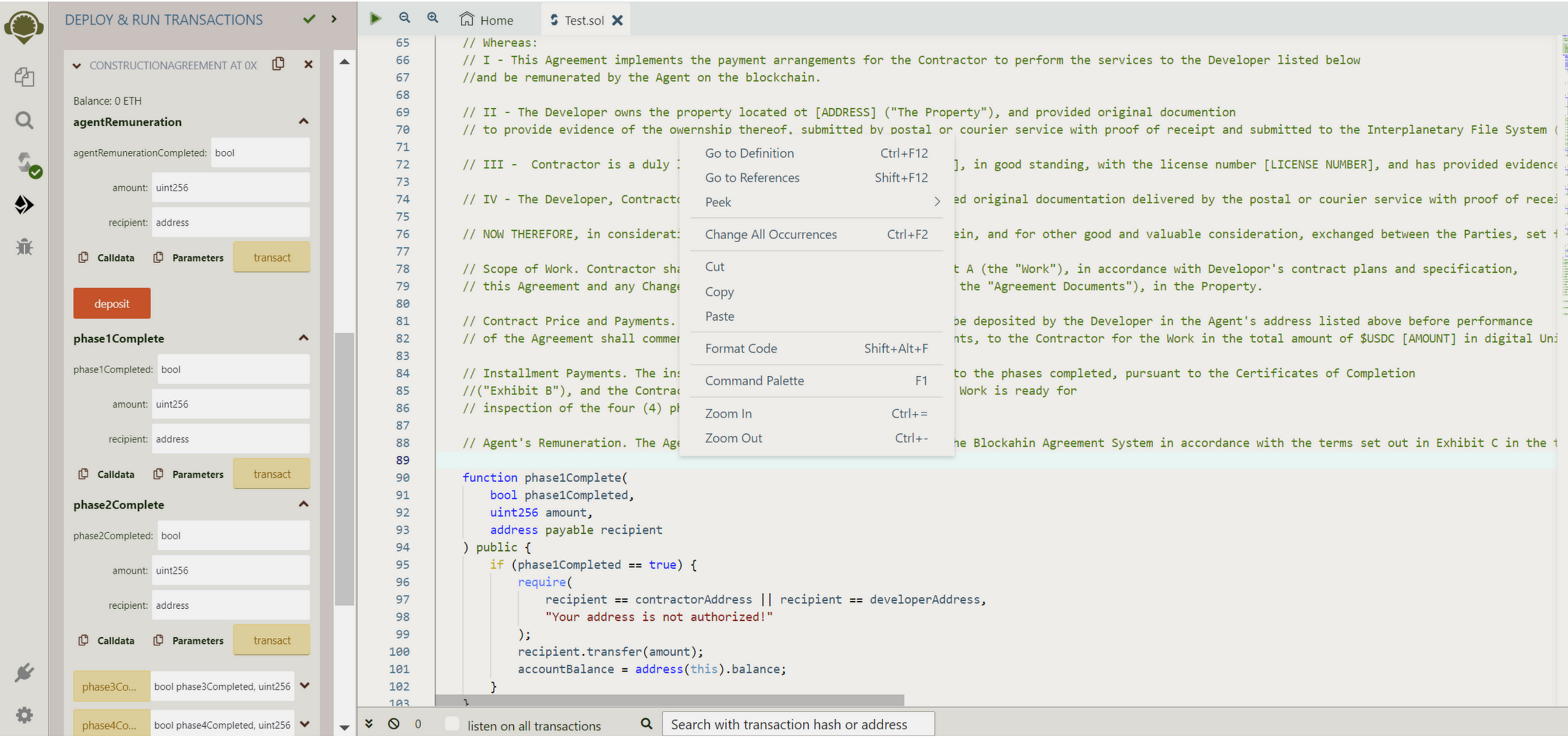
Imported 1 item

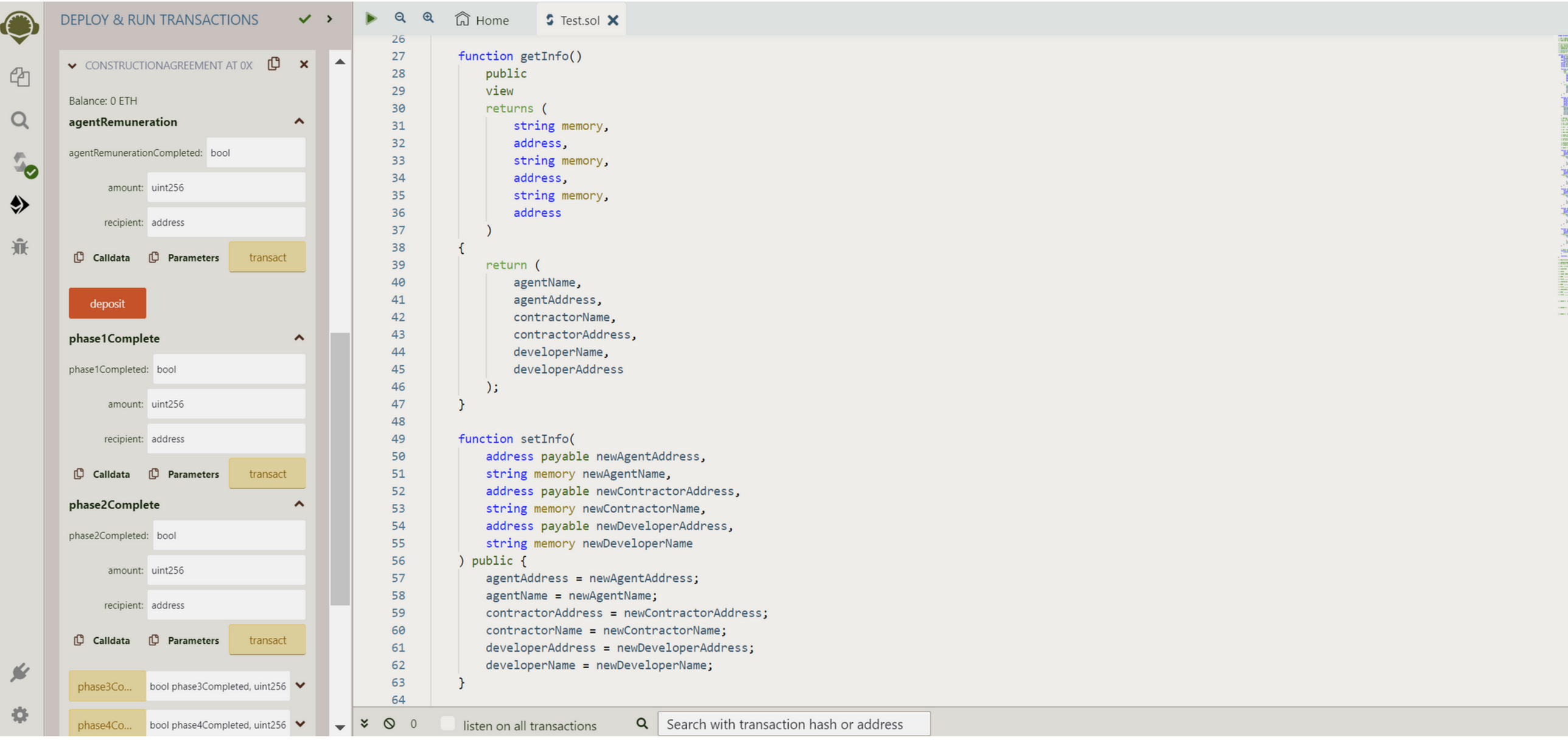
construction_contract_agreement.jpg.gpg | 37 KiB

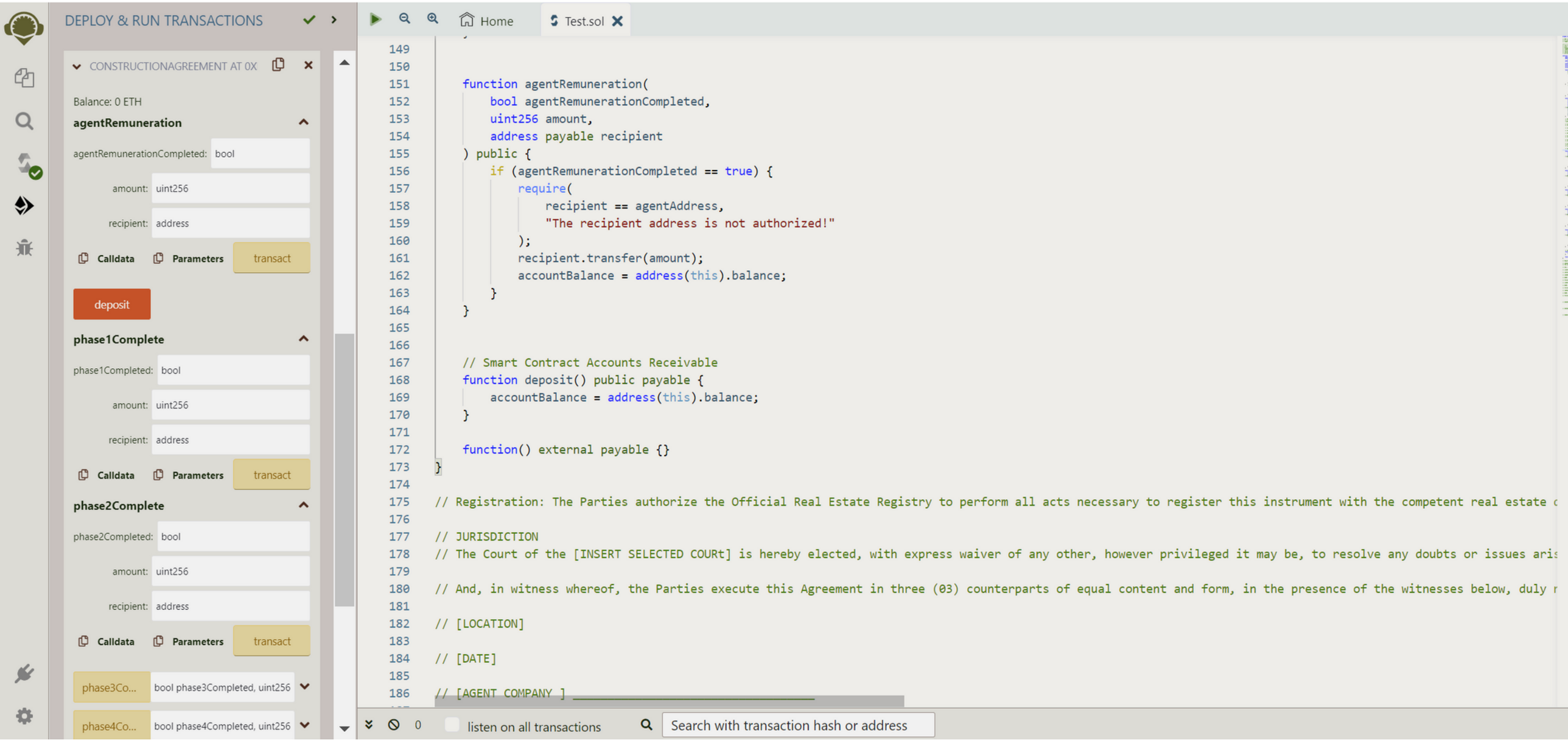
Revision 546fb5a
See the code
Report a bug

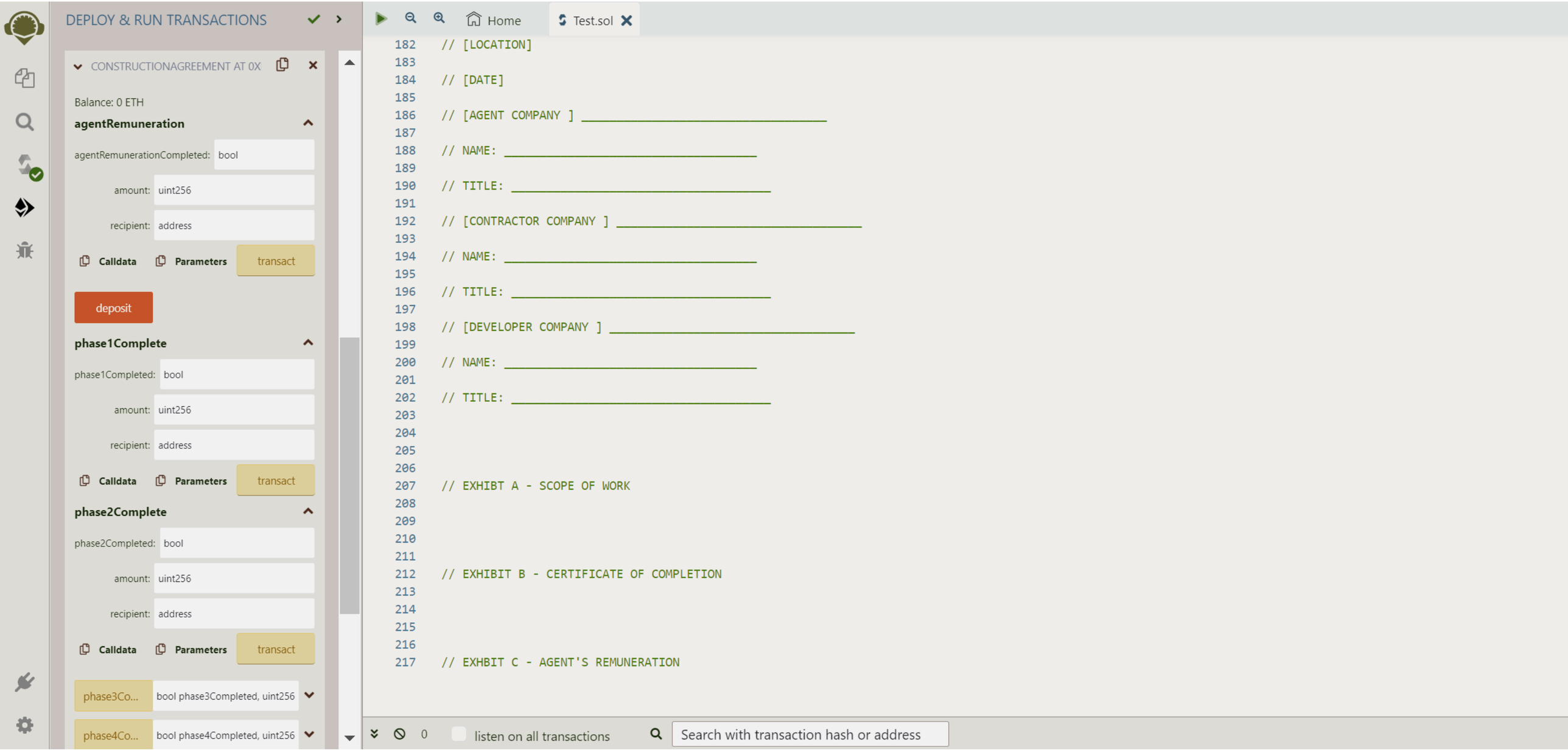
```
Russell@DESKTOP-JL9AU1F MINGW64 ~/Downloads
$ gpg --output construction_contract_agreement.jpg --decrypt construction_contract_agreement.jpg.gpg
```


DEPLOY & RUN TRANSACTIONS
CONSTRUCTIONAGREEMENT AT 0X
Balance: 0 ETH
agentRemuneration
agentRemunerationCompleted: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
deposit
phase1Complete
phase1Completed: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
phase2Complete
phase2Completed: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
phase3Co...
bool phase3Completed, uint256
phase4Co...
bool phase4Completed, uint256
Home
Test.sol
1 pragma solidity ^0.5.0;
2
3 // NOT A COMPLETE AGREEMENT.
4 // IT IS DRAFTED IN AN ATTEMPT TO EXPLAIN HOW AN AGREEMENT CAN BE EXECUTED AND PAYED FOR ON THE BLOCKCHAIN.
5 // THERE ARE MANY DIFFERENT PROVISIONS THAT NEED TO NEGOTIATED AND INCLUDED IN THE ACTUAL AGREEMENT.
6
7 //------------------- BLOCKCHAIN CONSTRUCTION AGREEMENT ---------------
8 // This Blockchain Construction Agreement ("Agreement") is hereby entered into on [DATE], by and between, the [AGENT COMPANY],
9 // duly represented in accordance with its [bylaws/articles of incorporation, articles of organization, etc.] ("Agent"),
10 // headquartered at [Address], duly represented in accordance with its [bylaws/articles of incorporation, articles of organization, etc.],
11 // [CONTRACTOR COMPANY], headquartered at [Address], duly represented in accordance with its [bylaws/articles of
12 // incorporation/articles of organization, etc.] and [DEVELOPER COMPANY], headquartered at [Address], duly represented in accordance with its
13 // [bylaws/articles of incorporation/articles or organization, etc.],
14 // pursuant to the following terms and conditions:
15
16 contract ConstructionAgreement {
17 address payable agentAddress = 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4;
18 address payable contractorAddress =
19 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db;
20 address payable developerAddress =
21 0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB;
22 uint256 public accountBalance;
23 string agentName = "Agent Company";
24 string contractorName = "Contractor Company";
25 string developerName = "Developer Company";
26
27 function getInfo()
28 public
29 view
30 returns (
0
listen on all transactions
Search with transaction hash or address
[vm] from: 0x5B3...eddC4 to: ConstructionAgreement.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0x4b7...12373
Debug


DEPLOY & RUN TRANSACTIONS
CONSTRUCTIONAGREEMENT AT 0X
Balance: 0 ETH
agentRemuneration
agentRemunerationCompleted: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
deposit
phase1Complete
phase1Completed: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
phase2Complete
phase2Completed: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
phase3Co...
bool phase3Completed, uint256
phase4Co...
bool phase4Completed, uint256
Home
Test.sol
// Whereas:
// I - This Agreement implements the payment arrangements for the Contractor to perform the services to the Developer listed below
//and be remunerated by the Agent on the blockchain.
// II - The Developer owns the property located ot [ADDRESS] ("The Property"), and provided original documention
// to provide evidence of the ownership thereof, submitted by postal or courier service with proof of receipt and submitted to the Interplanetary File System (
// III - Contractor is a duly
], in good standing, with the license number [LICENSE NUMBER], and has provided evidence
// IV - The Developer, Contracto
ed original documentation delivered by the postal or courier service with proof of recei
// NOW THEREFORE, in considerat
ein, and for other good and valuable consideration, exchanged between the Parties, set f
// Scope of Work. Contractor sha
t A (the "Work"), in accordance with Developor's contract plans and specification,
// this Agreement and any Change
the "Agreement Documents"), in the Property.
// Contract Price and Payments.
be deposited by the Developer in the Agent's address listed above before performance
// of the Agreement shall commer
nts, to the Contractor for the Work in the total amount of $USDC [AMOUNT] in digital Uni
// Installment Payments. The ins
to the phases completed, pursuant to the Certificates of Completion
//("Exhibit B"), and the Contrac
Work is ready for
// inspection of the four (4) ph
// Agent's Remuneration. The Age
he Blockahin Agreement System in accordance with the terms set out in Exhibit C in the t
function phase1Complete(
bool phase1Completed,
uint256 amount,
address payable recipient
) public {
if (phase1Completed == true) {
require(
recipient == contractorAddress || recipient == developerAddress,
"Your address is not authorized!"
);
recipient.transfer(amount);
accountBalance = address(this).balance;
}
Go to Definition Ctrl+F12
Go to References Shift+F12
Peek
Change All Occurrences Ctrl+F2
Cut
Copy
Paste
Format Code Shift+Alt+F
Command Palette F1
Zoom In Ctrl+=
Zoom Out Ctrl+-
0
listen on all transactions
Search with transaction hash or address


DEPLOY & RUN TRANSACTIONS
CONSTRUCTIONAGREEMENT AT 0X
Balance: 0 ETH
agentRemuneration
agentRemunerationCompleted: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
deposit
phase1Complete
phase1Completed: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
phase2Complete
phase2Completed: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
phase3Co...
bool phase3Completed, uint256
phase4Co...
bool phase4Completed, uint256
Home
Test.sol
26
27 function getInfo()
28 public
29 view
30 returns (
31 string memory,
32 address,
33 string memory,
34 address,
35 string memory,
36 address
37)
38 {
39 return (
40 agentName,
41 agentAddress,
42 contractorName,
43 contractorAddress,
44 developerName,
45 developerAddress
46);
47 }
48
49 function setInfo(
50 address payable newAgentAddress,
51 string memory newAgentName,
52 address payable newContractorAddress,
53 string memory newContractorName,
54 address payable newDeveloperAddress,
55 string memory newDeveloperName
56) public {
57 agentAddress = newAgentAddress;
58 agentName = newAgentName;
59 contractorAddress = newContractorAddress;
60 contractorName = newContractorName;
61 developerAddress = newDeveloperAddress;
62 developerName = newDeveloperName;
63 }
64
0
listen on all transactions
Search with transaction hash or address


DEPLOY & RUN TRANSACTIONS
CONSTRUCTIONAGREEMENT AT 0X
Balance: 0 ETH
agentRemuneration
agentRemunerationCompleted: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
deposit
phase1Complete
phase1Completed: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
phase2Complete
phase2Completed: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
phase3Co...
bool phase3Completed, uint256
phase4Co...
bool phase4Completed, uint256
Home
Test.sol
149
150
151 function agentRemuneration(
152 bool agentRemunerationCompleted,
153 uint256 amount,
154 address payable recipient
155) public {
156 if (agentRemunerationCompleted == true) {
157 require(
158 recipient == agentAddress,
159 "The recipient address is not authorized!"
160);
161 recipient.transfer(amount);
162 accountBalance = address(this).balance;
163 }
164 }
165
166
167 // Smart Contract Accounts Receivable
168 function deposit() public payable {
169 accountBalance = address(this).balance;
170 }
171
172 function() external payable {}
173 }
174
175 // Registration: The Parties authorize the Official Real Estate Registry to perform all acts necessary to register this instrument with the competent real estate
176
177 // JURISDICTION
178 // The Court of the [INSERT SELECTED COURt] is hereby elected, with express waiver of any other, however privileged it may be, to resolve any doubts or issues aris
179
180 // And, in witness whereof, the Parties execute this Agreement in three (03) counterparts of equal content and form, in the presence of the witnesses below, duly
181
182 // [LOCATION]
183
184 // [DATE]
185
186 // [AGENT COMPANY]
0
listen on all transactions
Search with transaction hash or address


DEPLOY & RUN TRANSACTIONS
CONSTRUCTIONAGREEMENT AT 0X
Balance: 0 ETH
agentRemuneration
agentRemunerationCompleted: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
deposit
phase1Complete
phase1Completed: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
phase2Complete
phase2Completed: bool
amount: uint256
recipient: address
Calldata
Parameters
transact
phase3Co...
bool phase3Completed, uint256
phase4Co...
bool phase4Completed, uint256
Home
Test.sol
182 // [LOCATION]
183
184 // [DATE]
185
186 // [AGENT COMPANY] _____
187
188 // NAME: _____
189
190 // TITLE: _____
191
192 // [CONTRACTOR COMPANY] _____
193
194 // NAME: _____
195
196 // TITLE: _____
197
198 // [DEVELOPER COMPANY] _____
199
200 // NAME: _____
201
202 // TITLE: _____
203
204
205
206
207 // EXHIBT A - SCOPE OF WORK
208
209
210
211
212 // EXHIBIT B - CERTIFICATE OF COMPLETION
213
214
215
216
217 // EXHBIT C - AGENT'S REMUNERATION
0
listen on all transactions
Search with transaction hash or address

ERC20 ERC721 ERC1155 **Governor** Custom

Copy to Clipboard | Open in Remix | Download

SETTINGS

Name

MyGovernor

Voting Delay | Voting Period

1 block | 1 week

1 block = 12 seconds

Proposal Threshold

0

Quorum % #

4

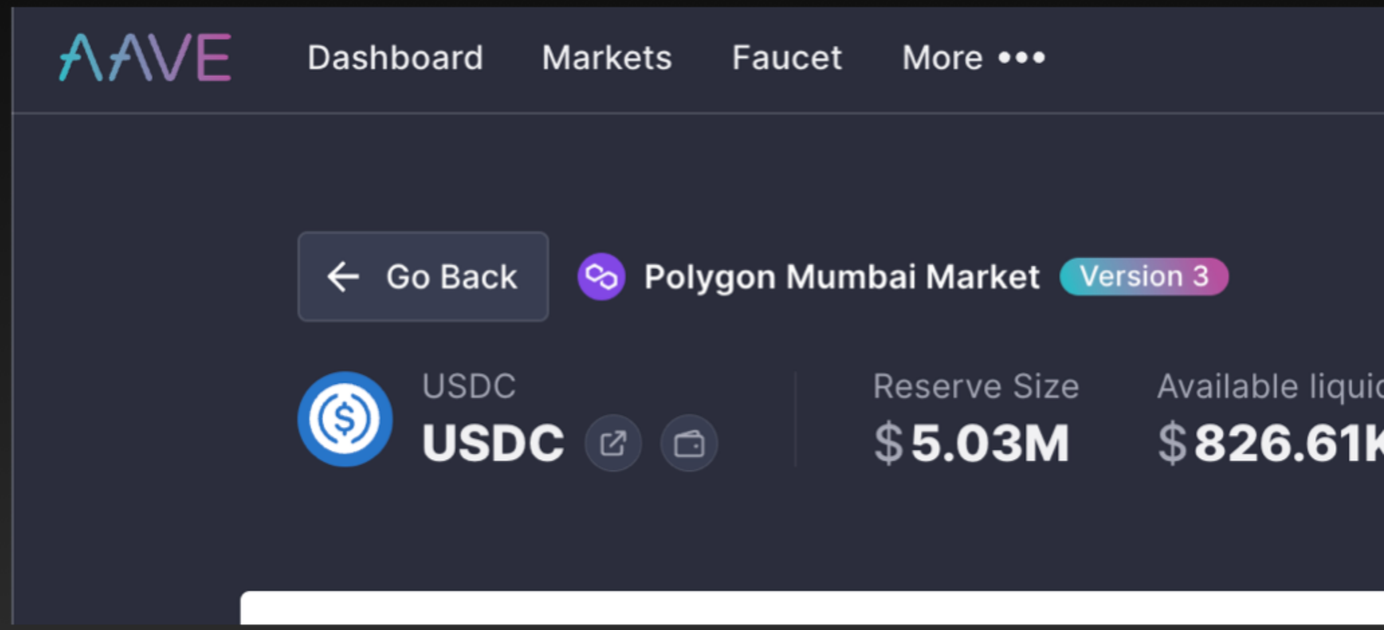
```
import "@openzeppelin/contracts/governance/Governor.sol";
import "@openzeppelin/contracts/governance/extensions/GovernorSettings.sol";
import "@openzeppelin/contracts/governance/extensions/GovernorCountingSimple.sol";
import "@openzeppelin/contracts/governance/extensions/GovernorVotes.sol";
import "@openzeppelin/contracts/governance/extensions/GovernorVotesQuorumFraction.sol";
import "@openzeppelin/contracts/governance/extensions/GovernorTimelockControl.sol";

contract MyGovernor is Governor, GovernorSettings, GovernorCountingSimple, GovernorVotes, GovernorVote
    constructor(IVotes _token, TimelockController _timelock)
        Governor("MyGovernor")
        GovernorSettings(1 /* 1 block */, 50400 /* 1 week */, 0)
        GovernorVotes(_token)
        GovernorVotesQuorumFraction(4)
        GovernorTimelockControl(_timelock)
    {}

    // The following functions are overrides required by Solidity.

    function votingDelay()
        public
```

COUNTER PARTY RISK PREVENTION

USDC FOR FIAT BASED SMART CONTRACT

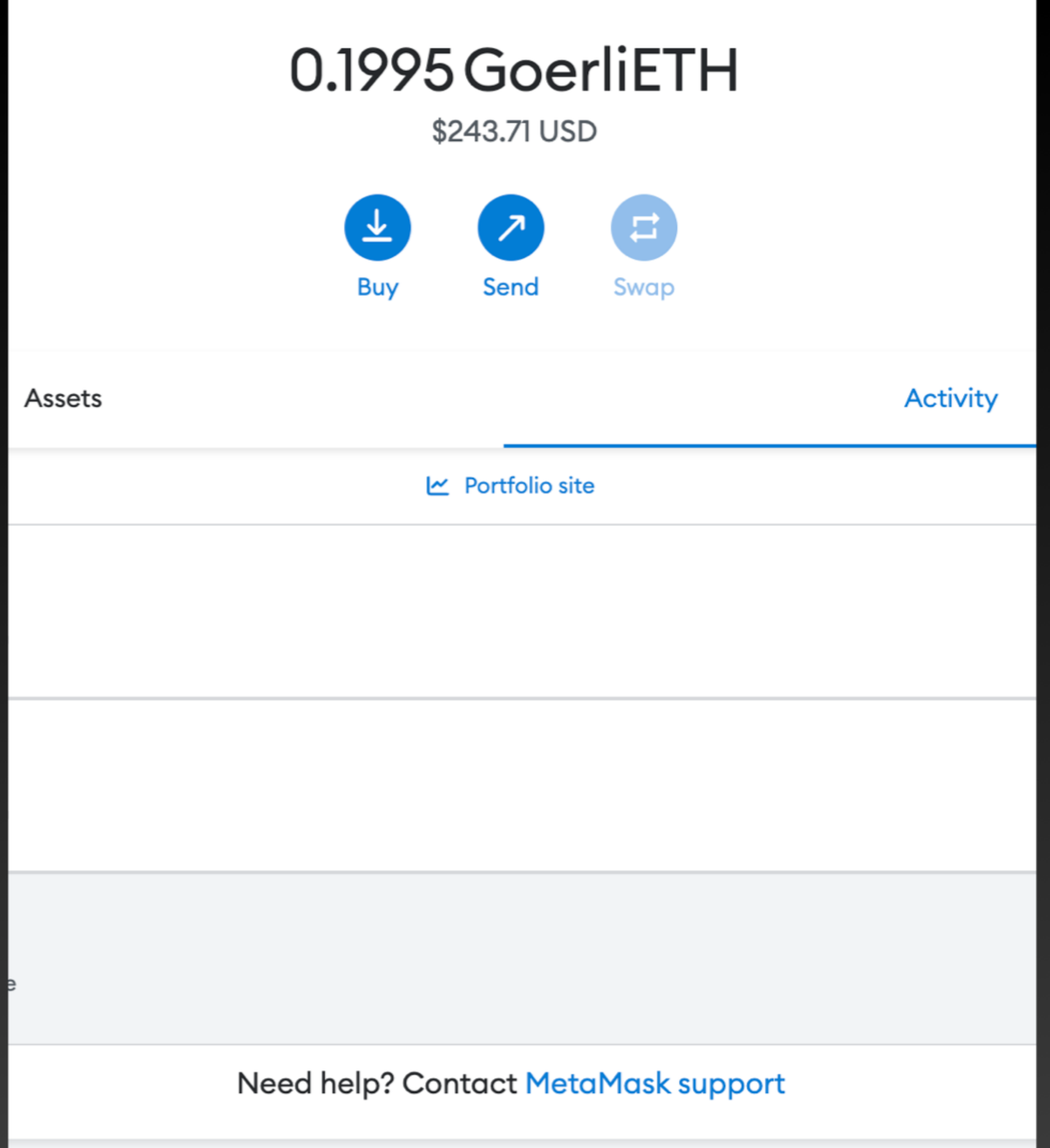
USDC
Reasons

- Stable US dollar coin that works with a smart contract for automated payments.
- Transparency
- Works with smart contracts



Challenges
With testing and using

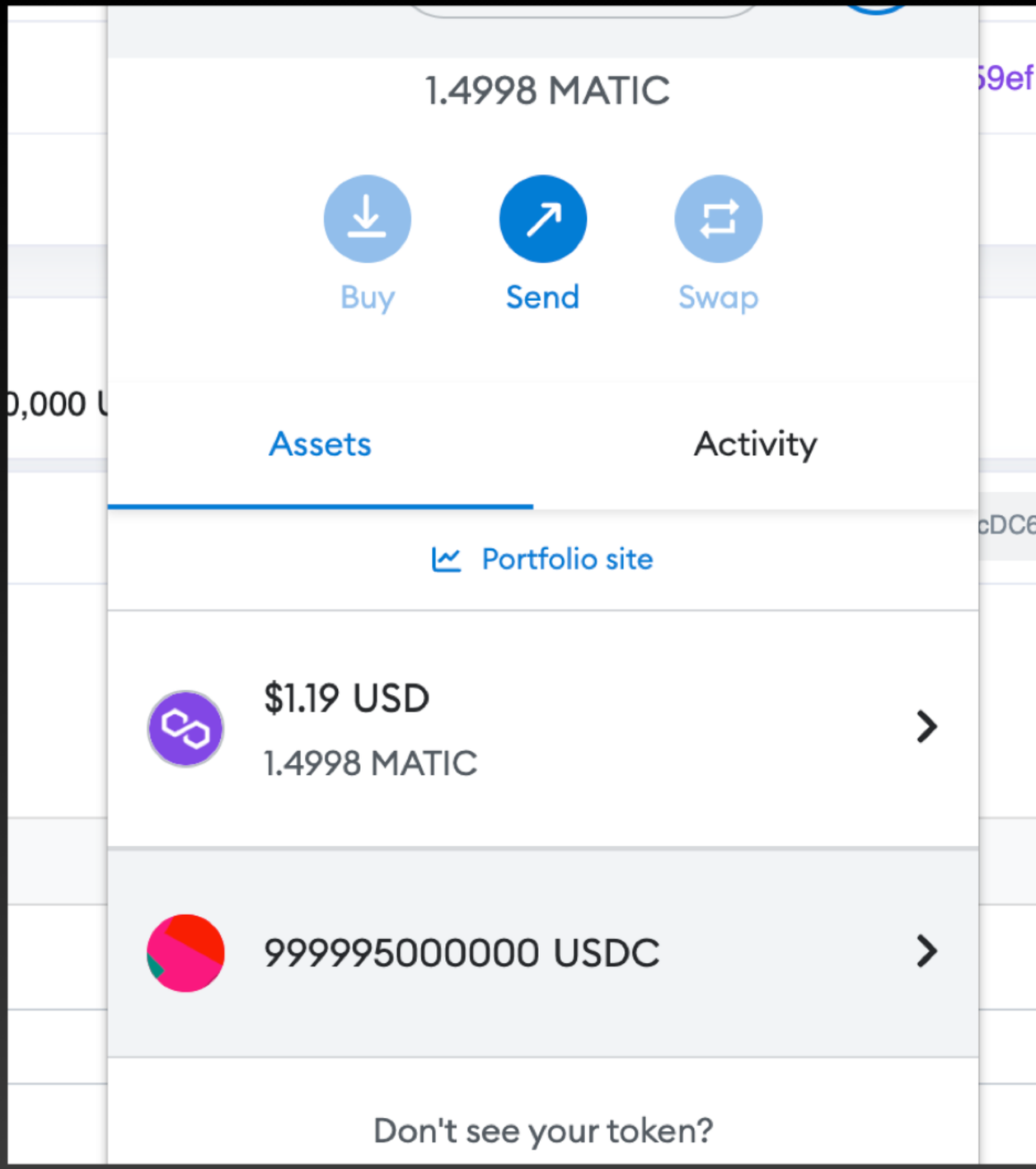
- Very new and there is not a seamless virtual environment.
- Has to be minted into an account.
- Ether or other test coins are not unlimited

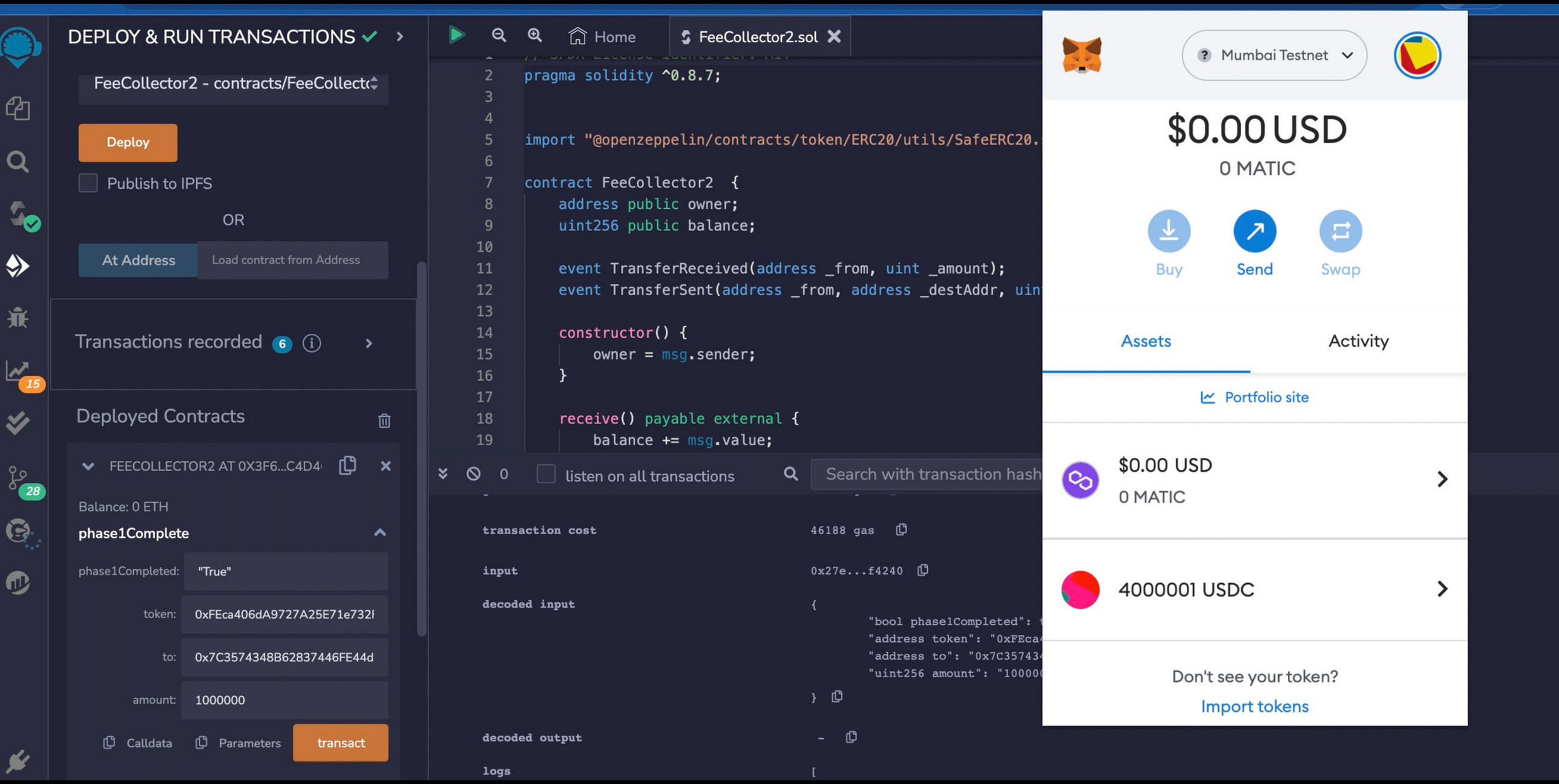


Accomplishments

Is USDC viable?

- Minted millions of USDC into a testing account. (So much trial and error, first)
- Transfered USDC into accounts deployed in Remix
- Contract functions work within Remix/Metamask and on the test network




DEPLOY & RUN TRANSACTIONS
FeeCollector2 - contracts/FeeCollect
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 6
Deployed Contracts
FEECOLLECTOR2 AT 0X3F6...C4D4
Balance: 0 ETH
phase1Complete
phase1Completed: "True"
token: 0xFEca406dA9727A25E71e732
to: 0x7C3574348B62837446FE44d
amount: 1000000
Calldata
Parameters
transact
Home
FeeCollector2.sol
pragma solidity ^0.8.7;
import "@openzeppelin/contracts/token/ERC20/utils/SafeERC20.
contract FeeCollector2 {
address public owner;
uint256 public balance;
event TransferReceived(address _from, uint _amount);
event TransferSent(address _from, address _destAddr, uin
constructor() {
owner = msg.sender;
}
receive() payable external {
balance += msg.value;
0
listen on all transactions
Search with transaction hash
transaction cost
46188 gas
input
0x27e...f4240
decoded input
"bool phase1Completed":
"address token": "0xFEca
"address to": "0x7C3574
"uint256 amount": "100000
}
decoded output
-
logs
[
Mumbai Testnet
$0.00 USD
0 MATIC
Buy
Send
Swap
Assets
Activity
Portfolio site
$0.00 USD
0 MATIC
4000001 USDC
Don't see your token?
Import tokens

Transfer E R C20

-$0.00 USD

Nov 23 · remix.ethereum.org

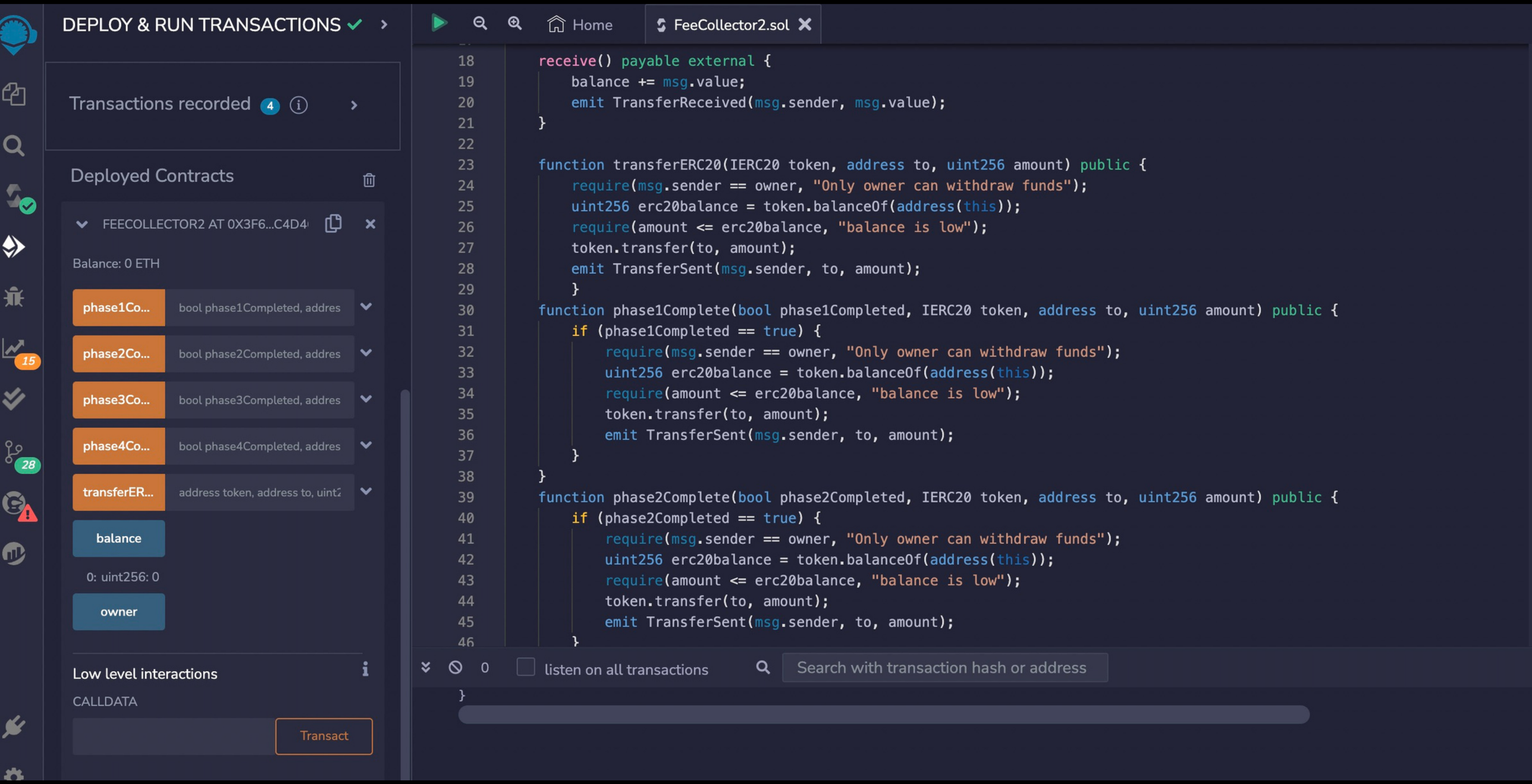
-0 MATIC

Contract interaction

-$0.00 USD

Nov 23 · remix.ethereum.org

-0 MATIC


DEPLOY & RUN TRANSACTIONS
Transactions recorded 4
Deployed Contracts
FEECOLLECTOR2 AT 0X3F6...C4D4
Balance: 0 ETH
phase1Co... bool phase1Completed, addres
phase2Co... bool phase2Completed, addres
phase3Co... bool phase3Completed, addres
phase4Co... bool phase4Completed, addres
transferER... address token, address to, uint2
balance
0: uint256: 0
owner
Low level interactions
CALLDATA
Transact
Home
FeeCollector2.sol
receive() payable external {
balance += msg.value;
emit TransferReceived(msg.sender, msg.value);
}
function transferERC20(IERC20 token, address to, uint256 amount) public {
require(msg.sender == owner, "Only owner can withdraw funds");
uint256 erc20balance = token.balanceOf(address(this));
require(amount <= erc20balance, "balance is low");
token.transfer(to, amount);
emit TransferSent(msg.sender, to, amount);
}
function phase1Complete(bool phase1Completed, IERC20 token, address to, uint256 amount) public {
if (phase1Completed == true) {
require(msg.sender == owner, "Only owner can withdraw funds");
uint256 erc20balance = token.balanceOf(address(this));
require(amount <= erc20balance, "balance is low");
token.transfer(to, amount);
emit TransferSent(msg.sender, to, amount);
}
}
function phase2Complete(bool phase2Completed, IERC20 token, address to, uint256 amount) public {
if (phase2Completed == true) {
require(msg.sender == owner, "Only owner can withdraw funds");
uint256 erc20balance = token.balanceOf(address(this));
require(amount <= erc20balance, "balance is low");
token.transfer(to, amount);
emit TransferSent(msg.sender, to, amount);
}
0
listen on all transactions
Search with transaction hash or address
}

1077 txIndex:1] **from:** 0x092...fc1Da **to:** FeeCollector.transferERC20(address,address,uint256) 0xD6c.
i data: 0x9db...a2000 **logs:** 0 **hash:** 0xf54...df08a

true Transaction mined and execution succeed

0x0e8a3ab0938886828b4f3ffbe18587c9e0577ef512644573b422fb9346762351

0x092197E7dAFcDC69F45C412e5a9f109A844fc1Da

FeeCollector.transferERC20(address,address,uint256) 0xD6c82a5aF8AAE37cc6FC8060E458BDa8A6035b73

21988 gas

21988 gas

0x9db...a2000


```
{
        "address token": "0xFEca406dA9727A25E71e732F9961F680059eF1F9",
        "address to": "0x7C3574348B62837446FE44da17f5cC5174A3fa8A",
        "uint256 amount": "2000000000000"
}
```


-

COUNTER PARTY RISK PREVENTION

MACHINE LEARNING FOR VALIDATION

C: > Users > Russell > Documents > Fintech-Workspace > Python_Project > FinalProject_Update > ConvertImage.py > ...

```python
from PIL import Image
import os, sys
import cv2
import numpy as np
from pathlib import Path

'''
Converts all images in a directory to '.npy' format.
Use np.save and np.load to save and load the images.
Use it for training your neural networks in ML/DL projects.
'''

# Path to image directory
path = os.path.join("C:",os.sep,"Users","Russell","Documents","Fintech-Workspace","Python_Project","FinalProject","Training")
dirs = os.listdir( path )
dirs.sort()
x_train=[]

def load_dataset():
    # Append images to a list
    for item in dirs:

        if item.endswith('jpg'):
            im = Image.open(path+"\\"+item).convert("RGB")
            im = np.array(im)
            x_train.append(im)

if __name__ == "__main__":

    load_dataset()

    # Convert and save the list of images in '.npy' format
    #imgset=np.array(x_train)
    np.save("train1.npy",x_train)
```

File Edit Selection View Go Run Terminal Help

Search

ConvertImage.py | CNN_Classification.ipynb

C: > Users > Russell > Documents > Fintech-Workspace > Python_Project > FinalProject_Update > CNN_Classification.ipynb > import keras

+ Code + Markdown | Run All | Clear Outputs of All Cells | Outline ···

base (Python 3.9.12)

```python
import keras
import keras.backend as k
from keras.models import Sequential
from keras.layers import Conv2D,MaxPooling2D,ZeroPadding2D
from keras.layers import Dense,Dropout,Flatten
from keras.layers.normalization import BatchNormalization
from keras import losses
from keras.optimizers import Adam,RMSprop,Adadelta
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelBinarizer
from keras.applications.vgg16 import preprocess_input,decode_predictions
import matplotlib.pyplot as plt
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import scipy as sp
import matplotlib.pyplot as plt
import seaborn as sns
import skimage.transform as st
```

[26] Python

```python
train_data = pd.read_csv('train_labels.csv')    # Load the train dataset
train_data1 = np.load('train1.npy',allow_pickle=True)
print(train_data.shape)
print(train_data1.shape)
train_data1
```

[30] Python

```
(130, 1)
(130,)

Output exceeds the size limit. Open the full output data in a text editor
array([array([[[49, 38, 32],
               [53, 42, 36],
               [46, 35, 29],
               ...,
               [42, 32, 31],
               [38, 27, 31],
               [40, 29, 33]],

              [[49, 38, 32],
               [55, 44, 38],
               [43, 32, 26],
```

ConvertImage.py | CNN_Classification.ipynb

C: > Users > Russell > Documents > Fintech-Workspace > Python_Project > FinalProject_Update > CNN_Classification.ipynb > import keras

+ Code + Markdown | Run All | Clear Outputs of All Cells | Outline ··· base (Python 3.9.12

```
test_data = pd.read_csv('test_labels.csv')     # Load the test dataset
test_data1 = np.load('test.npy',allow_pickle=True)
print(test_data.shape)
print(test_data1.shape)
```

[6] Python

```
(130, 1)
(130,)
```

```
#Split the train dataset
X_train,X_val,Y_train,Y_val=train_test_split(train_data,train_data1,test_size=0.2,random_state=431)
print(X_train.shape,X_val.shape)
print(Y_train.shape,Y_val.shape)
```

[7] Python

```
(104, 1) (26, 1)
(104,) (26,)
```

```
num_classes=2

#CNN model
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',input_shape=(32,32,3)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
```

[9] Python

```
WARNING:tensorflow:From C:\Users\Russell\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:4070: The name tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2d instead.
```

Compile Model

```
model.compile(loss=losses.categorical_crossentropy,optimizer=Adadelta(),metrics=['accuracy'])
```

[10] Python

⊗ 0 ⚠ 2 Jupyter Server: Local Cell 1 of 10

File Edit Selection View Go Run Terminal Help

Search

ConvertImage.py | CNN_Classification.ipynb

C: > Users > Russell > Documents > Fintech-Workspace > Python_Project > FinalProject_Update > CNN_Classification.ipynb > import keras

+ Code + Markdown | Run All | Clear Outputs of All Cells | Outline ···

base (Python 3.9.12)

Training the Model

[11]

```
cnn=model.fit(X_train,Y_train,batch_size=128,epochs=50,verbose=1,validation_data=(X_val,Y_val),shuffle=True)
```

Python

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_19780\3634122649.py in <module>
----> 1 cnn=model.fit(X_train,Y_train,batch_size=128,epochs=50,verbose=1,validation_data=(X_val,Y_val),shuffle=True)

~\Anaconda3\lib\site-packages\keras\engine\training.py in fit(self, x, y, batch_size, epochs, verbose, callbacks, validation_split, validation_data, shuffle, class_weight, sample_weight, initial_epoch, steps_per_epoch, validation_s
teps, validation_freq, max_queue_size, workers, use_multiprocessing, **kwargs)
   1152             sample_weight=sample_weight,
   1153             class_weight=class_weight,
-> 1154             batch_size=batch_size)
   1155
   1156         # Prepare validation data.

~\Anaconda3\lib\site-packages\keras\engine\training.py in _standardize_user_data(self, x, y, sample_weight, class_weight, check_array_lengths, batch_size)
    577             feed_input_shapes,
    578             check_batch_axis=False,  # Don't enforce the batch size.
--> 579             exception_prefix='input')
    580
    581         if y is not None:

~\Anaconda3\lib\site-packages\keras\engine\training_utils.py in standardize_input_data(data, names, shapes, check_batch_axis, exception_prefix)
    133                         ': expected ' + names[i] + ' to have ' +
    134                         str(len(shape)) + ' dimensions, but got array '
--> 135                         'with shape ' + str(data_shape))
    136                 if not check_batch_axis:
    137                     data_shape = data_shape[1:]

ValueError: Error when checking input: expected conv2d_1_input to have 4 dimensions, but got array with shape (104, 1)
```

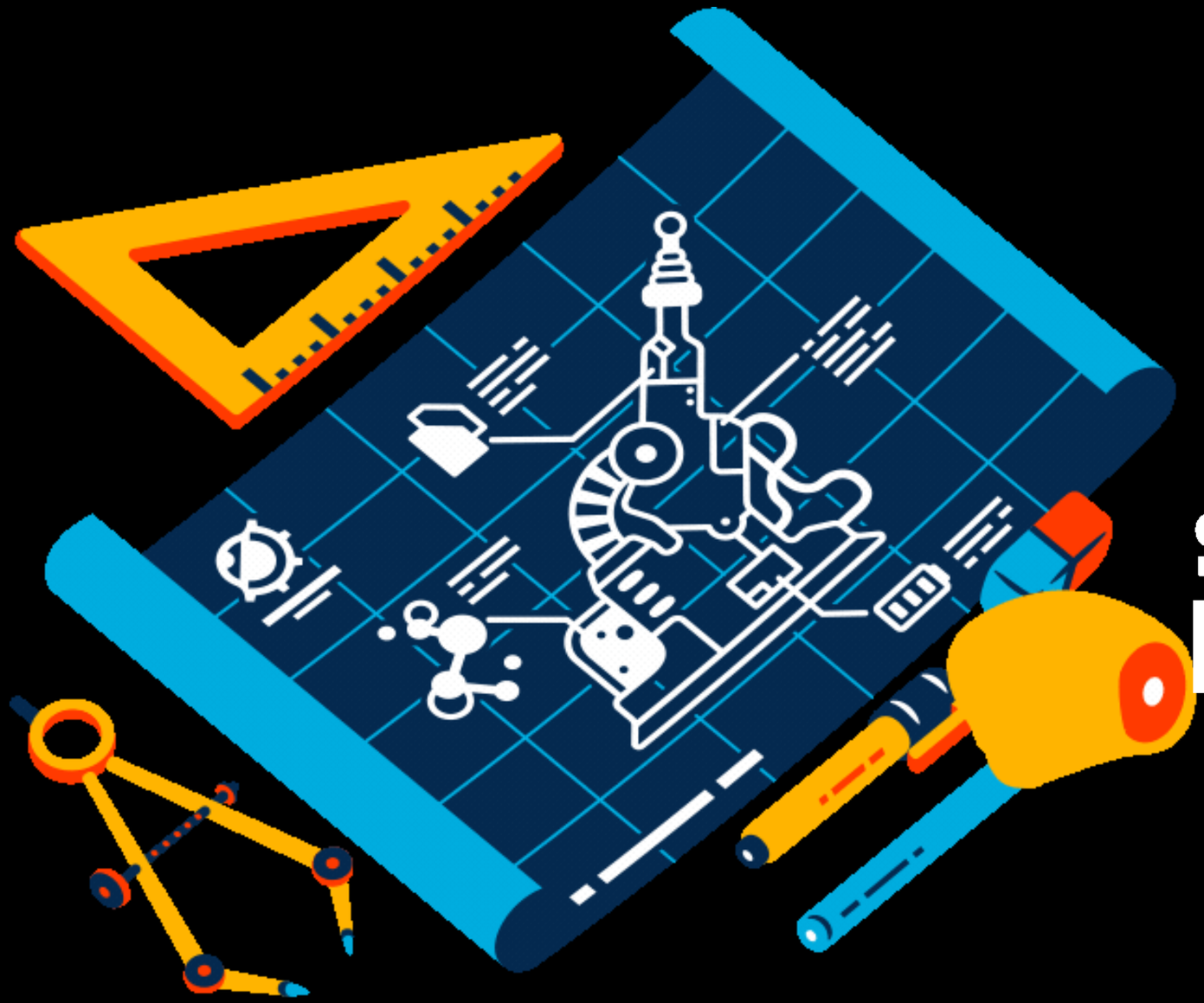
[ ]

Python

COUNTER PARTY RISK PREVENTION

FRONT END INTERFACE


02587501569787
156789147
369852147
258456159
741258963
DISCOVER
40 %
60 %
80 %



SIGN UP
NOW

Upload Your Invoice

Currency should be defined in USD

Select Constractor ID

Contract ID

Select Milestone:

First Milestone

Pick a date

2022/11/23

Invoice Number

Invoice Amount

0.00 − +

Upload Your invoice

Drag and drop files here

Limit 200MB per file

Browse files

We make construction process painless

Upload project milestone Invoice

Contract ID

Type Your Name Here

Your Email

Message should include your Invoice number

Send

Here to make your payments promptly

For Offical Use Only

Select an Invoice to Pay

1234

Select Constractor ID

Contract ID

Select Account Number

Select Account

Payment Date

2022/11/23

Enter Invoice Amount

0.00 − +

Pay Invoice

Construction Expense and Payment tracker

A risk free way of executing construction contracts

Contract Milestones | **Transaction History**

Account Tracking

Select Account Number

0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db

Cntract ID

AB276NYC

Payment Date

2022/11/23

Check Payment

	expense	key	milestone	accountNumber	contractId	date	paymentAmount
2	2900000	234567	First Milestone	0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db	AB276NYC	2022-11-09	200000
0	200000	1234	Second Milestone	0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db	AB276NYC	2022-11-23	600000
1	2000000	234500	First Milestone	0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db	AB276NYC	2022-11-23	500000

Project Conclusion

Because exploits can leave a DAO drained of millions of dollars of its assets. We prioritized the highest security; we use OpenZeppelin to create our DOA.

OpenZeppelin provides security products to build, automate, and operate decentralized applications. We also protect leading organizations by performing security audits on their systems.

Project Conclusion

DAOs rely heavily on smart contracts. These logically coded agreements dictate decision-making based on underlying activity on a blockchain.

Based on the outcome of a decision, certain code may be implemented to increase the circulating supply, burn of a select amount of reserve tokens, or issue preferred rewards to parties with the smart contract for our clients.

Project Conclusion

Contractors can make proposals to the DAO on payment terms and distribution and the Developer negotiates terms and/ or accepts the proposals.

The DOA writes the smart contracts per project with mention deliverables, terms and conditions.

Project Conclusion

There will be an initial funding period, in which people add funds to the DAO by purchasing tokens that represent ownership by the Developer– a crowd sale or an initial coin offering (ICO) – to give it the resources it needs. When the funding period is over, the DAO begins to operate and the contractor start phase of the project.

All transactions and activity through the DAO are posted on a blockchain, making all actions of stakeholders publicly viewable.

COLUMBIA | ENGINEERING
The Fu Foundation School of Engineering and Applied Science

PROJECT THREE

SMART CONTRACTS FOR CONSTRUCTION

Developed By
Crystal Evans, Patrick Ruejoma, Jeff Smith, Mike Cutno, Russell Moore, Sheng Gao