KYC Contract

Problem Proposed: KYC (Know Your Customer) is a service provided by financial institutions such as banks. There are both public and private sector banks managed by a central bank. These banks are banned by the central bank from adding any new customer and do any more customer KYCs as they see suspicious activities that need to be sorted out first. Despite this, the banks add new customers and do the KYC in the background.

Solution Proposed: An immutable solution is needed where the central bank maintains a list of all the banks and tracks which banks are allowed to add new customers and perform KYC. It can also track which customer KYC is completed or pending along with customer details.

Banks can also add the new customer if allowed and do the KYC of the customers.

Approach

My setup for the project:

1. Understanding problem statement
2. Correlating data structures and exception handling to use
3. Implement smart contract in Solidity (using Remix)
4. Perform unit-testing on local blockchain or Ganache
5. Deploy to test-net

Smart Contract

I am using the following data structures in my smart contract:

```
// given Bank will have these properties
    struct Bank {
        string bankName;
        address bankAddress;
        uint256 kycCount;
        bool canAddUser;
        bool kycPrivilege;
    }
// given Customer will have these properties
struct Customer {
    string customerName;
    string customerData;
    address customerBank;
    bool kycStatus;
}
// bank address => bank type
mapping(address => Bank) public banks;
// customer address => customer type
mapping(string => Customer) public customers;
```

NOTE: only admin can add banks and only banks can add customers to their bank

Exception Handling

1. OnlyOwner modifier by OpenZeppelin
2. Checking if msg.sender == address of bank for bank functions

NOTE: this is tested using brownie / pytest (logs available below)

Tests

1. test_can_add_bank_to_blockchain: This tests that only owner can add a bank and whether the bank is added successfully
2. test_can_block_bank_from_adding_user: This tests that whether blocking adding is working successfully
3. test_can_block_bank_from_kyc_priviledge: This tests that whether blocking kyc is working successfully
4. test_can_add_customer_to_bank: This tests that only bank can add a customer and can't add after blocking and whether customer is added successfully
5. test_perform_kyc: This tests that status is changed after performing kyc and can only bank has permission for this

Brownie Framework

1. Deploying to Rinkeby

Logs

```
brownie run scripts/deploy.py --network rinkeby
Brownie v1.18.1 - Python development framework for Ethereum
Project is the active project.
Running 'scripts/deploy.py::main'...
Enter password for "shivam":
Transaction sent: 0x6f1d34d2fa401f0b7bd254732e0237d7c56a1a208ed78ee915852597a0
Gas price: 1.000000615 gwei Gas limit: 2027669 Nonce: 71
KYC.constructor confirmed Block: 10931024 Gas used: 1843336 (90.91%)
KYC deployed at: 0x43226D915358da7B2432A22D38a1632fbB683284
Contract address: 0x43226D915358da7B2432A22D38a1632fbB683284
Enter password for "shivam":
Adding bank to contract
Transaction sent: 0x8a3cc360199e3fb2c773b0f0a150c9f068f0ca0a40c0990885d2e3a3d9
Gas price: 1.000000615 gwei Gas limit: 103557 Nonce: 72
KYC.addNewBankToBlockchain confirmed Block: 10931025 Gas used: 94143 (90.91%)
KYC.addNewBankToBlockchain confirmed Block: 10931025 Gas used: 94143 (90.91%)
Bank added to contract
Enter password for "shivam":
Adding bank to contract
Transaction sent: 0x257a079c8a3cc5bd967b7155b9942b6983f2ec641e8a79abd11dacd09a
Gas price: 1.000000616 gwei Gas limit: 45832 Nonce: 73
KYC.addNewBankToBlockchain confirmed Block: 10931026 Gas used: 39480 (86.14%)
KYC.addNewBankToBlockchain confirmed Block: 10931026 Gas used: 39480 (86.14%)
Bank added to contract
Adding customer to bank
Transaction sent: 0x2e7269313a046563971a1c227717078d3603ad2cfca876f6c7039cc073
```

```
Gas price: 1.000000616 gwei Gas limit: 138727 Nonce: 74
KYC.addNewCustomerToBank confirmed Block: 10931027 Gas used: 123935 (89.34%)
KYC.addNewCustomerToBank confirmed Block: 10931027 Gas used: 123935 (89.34%)
Customer John added to bank: 0xA37a0eE21f5964B27fD577002Ed93e75d3357244
Adding customer to bank
Transaction sent: 0x4065c1de9234845b8509338d981729469b86a2e73f6dbbda2d405fca82
Gas price: 1.000000616 gwei Gas limit: 119891 Nonce: 75
KYC.addNewCustomerToBank confirmed Block: 10931028 Gas used: 106811 (89.09%)
KYC.addNewCustomerToBank confirmed Block: 10931028 Gas used: 106811 (89.09%)
Customer Bob added to bank: 0xA37a0eE21f5964B27fD577002Ed93e75d3357244
KYC status of John: False
KYC status of Bob: False
Performing KYC of John on bank1
Transaction sent: 0x70da944bc804c77b6f8fc6af0a3f3e5e45895d7d0ff39653e5828991a6
Gas price: 1.000000615 gwei Gas limit: 45717 Nonce: 76
KYC.performKycOfCustomer confirmed Block: 10931029 Gas used: 41561 (90.91%)
KYC status of John: False
Performing KYC of Bob on bank2
Transaction sent: 0xb559a9d0571a0873d5e6d23d7d5b19f71955b655745fb2c4c0fcf38bc1
Gas price: 1.000000615 gwei Gas limit: 45703 Nonce: 77
KYC.performKycOfCustomer confirmed Block: 10931030 Gas used: 41549 (90.91%)
KYC status of Bob: True
```

1. Testing

Logs

```
============================================ test session starts ===============
platform darwin -- Python 3.9.5, pytest-6.2.5, py-1.11.0, pluggy-1.0.0
rootdir: /Users/shivamarora/Downloads/Projects/Freelance/Blockchain/project
plugins: eth-brownie-1.18.1, forked-1.4.0, web3-5.27.0, xdist-1.34.0, hypothes
collected 5 items
Launching 'ganache-cli --accounts 10 --hardfork istanbul --gasLimit 12000000 -
tests/test_kyc.py ..... [100%]
============================================= 5 passed in 8.53s ================
Terminating local RPC client...
```