**KYC System**

Additional feature implemented:

We have added a state variable **banksVotedForCustomer**. **It tracks whether a bank has voted or not for a customer.** Its mapping with key of customer username which returns another mapping of bank’s address as key and a bool assigned to it. This property can verify and restrict a bank to vote only once for a specific customer.

Variable for testing:

adminAddress: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

Bank Addresses:  
bankAddr1: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2   
bankAddr2: 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db  
bankAddr3: 0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB  
bankAddr4: 0x617F2E2fD72FD9D5503197092aC168c91465E7f2  
bankAddr5: 0x17F6AD8Ef982297579C203069C1DbfFE4348c372  
bankAddr6: 0x5c6B0f7Bf3E7ce046039Bd8FABdfD3f9F5021678   
bankAddr7: 0x03C6FcED478cBbC9a4FAB34eF9f40767739D1Ff7  
bankAddr8: 0x1aE0EA34a72D944a8C7603FfB3eC30a6669E454C  
bankAddr9: 0x0A098Eda01Ce92ff4A4CCb7A4fFFb5A43EBC70DC  
bankAddr10: 0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c  
bankAddr11: 0x14723A09ACff6D2A60DcdF7aA4AFf308FDDC160C  
bankAddr12: 0x4B0897b0513fdC7C541B6d9D7E929C4e5364D2dB

Customer usernames  
customer1 (added by bank1): 0x637573746f6d65725f3100000000000000000000000000000000000000000000   
customer2 (added by bank2): 0x637573746f6d65725f3100000000000000000000000000000000000000000000

**Add Banks to the KYC System**.

Once the contract is deployed, by the admin, add banks with following input sequence.

Function: addBank (bytes32 \_name, address \_ethAddress, bytes32 \_regNumber)

Bank 1 (bankAddr1)

0x42616e6b20310000000000000000000000000000000000000000000000000000, 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2, 0x3100000000000000000000000000000000000000000000000000000000000000

Bank 2 (bankAddr2)

0x42616e6b20320000000000000000000000000000000000000000000000000000, 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db, 0x3200000000000000000000000000000000000000000000000000000000000000

Bank 3 (bankAddr3)

0x42616e6b20330000000000000000000000000000000000000000000000000000, 0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB, 0x3300000000000000000000000000000000000000000000000000000000000000

Bank 4 (bankAddr4)

0x42616e6b20340000000000000000000000000000000000000000000000000000, 0x617F2E2fD72FD9D5503197092aC168c91465E7f2, 0x3400000000000000000000000000000000000000000000000000000000000000

Bank 5 (bankAddr5)

0x42616e6b20350000000000000000000000000000000000000000000000000000, 0x17F6AD8Ef982297579C203069C1DbfFE4348c372, 0x3500000000000000000000000000000000000000000000000000000000000000

Bank 6 (bankAddr6)

0x42616e6b20360000000000000000000000000000000000000000000000000000, 0x5c6B0f7Bf3E7ce046039Bd8FABdfD3f9F5021678, 0x3600000000000000000000000000000000000000000000000000000000000000

Bank 7 (bankAddr7)

0x42616e6b20370000000000000000000000000000000000000000000000000000, 0x03C6FcED478cBbC9a4FAB34eF9f40767739D1Ff7, 0x3700000000000000000000000000000000000000000000000000000000000000

Bank 8 (bankAddr8)

0x42616e6b20380000000000000000000000000000000000000000000000000000, 0x1aE0EA34a72D944a8C7603FfB3eC30a6669E454C, 0x3800000000000000000000000000000000000000000000000000000000000000

Bank 9 (bankAddr9)

0x42616e6b20390000000000000000000000000000000000000000000000000000, 0x0A098Eda01Ce92ff4A4CCb7A4fFFb5A43EBC70DC, 0x3900000000000000000000000000000000000000000000000000000000000000

Bank 10 (bankAddr10)

0x42616e6b20313000000000000000000000000000000000000000000000000000, 0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c, 0x3130000000000000000000000000000000000000000000000000000000000000

Bank 11 (bankAddr11)

0x42616e6b20313100000000000000000000000000000000000000000000000000, 0x14723A09ACff6D2A60DcdF7aA4AFf308FDDC160C, 0x3131000000000000000000000000000000000000000000000000000000000000

Bank 12 (bankAddr12)

0x42616e6b20313200000000000000000000000000000000000000000000000000, 0x4B0897b0513fdC7C541B6d9D7E929C4e5364D2dB, 0x3132000000000000000000000000000000000000000000000000000000000000

Any attempt to add the same bank say Bank10 shall fail, as Bank already exist.

**Add Customers and KYC request to KYC System**

Functions:

* addCustomer (bytes32 \_name, bytes32 \_data)
* addKycRequest (bytes32 \_cUsername, bytes32 \_customerData)

Customer 1 (0x637573746f6d65725f3100000000000000000000000000000000000000000000):

Switch account to Bank 1 and add Customer for Bank 1 with following inputs followed by making a KYC request with same inputs

0x637573746f6d65725f3100000000000000000000000000000000000000000000, 0x637573746f6d65725f315f646174610000000000000000000000000000000000

Customer 2 (0x637573746f6d65725f3200000000000000000000000000000000000000000000):

Switch account to Bank 2 and add Customer for Bank 2 with following inputs followed by making a KYC request with same inputs

0x637573746f6d65725f3200000000000000000000000000000000000000000000, 0x637573746f6d65725f325f646174610000000000000000000000000000000000

**Upvote and Downvote Customers.**

Functions:

* upVoteCustomer (bytes32 \_cUsername)
* downVoteCustomer (bytes32 \_cUsername)

At this point the system holds two KYC requests. Switch accounts from Bank 1 to Bank 12 and cast the votes, either up or down. Attempts to cast vote by banks own customer will fail, reporting error as “Bank is not allowed to cast vote”. Attempts to vote are also limited to a valid customer with valid KYC request. Once done call the function viewCustomer to check the current KYC status of the customer. It would be marked as true if number of upvotes is grater than downvotes. Further it would be marked as false, if more than 1/3rd of the total banks has downvoted the customer.

For instance, upvote **Customer 1** with the username “**0x637573746f6d65725f3100000000000000000000000000000000000000000000**” 5 times followed downvoting 4 times. The KYC status would be marked as true in this case. Similarly, upvote **Customer 2** with the username “**0x637573746f6d65725f3100000000000000000000000000000000000000000000**” 7 times followed downvoting 6 times. The KYC status would be marked as false in this case, since more than 1/3rd of the banks has downvoted the customer.

**View Customer**

Function: viewCustomer (bytes32 \_cUsername)

The customer details including the results of KYC status as per prior operations can be confirmed using the viewCustomer function, by calling it with username of above customers (Customer 1 and Customer 2).

**Modify Customer**

Function: modifyCustomer (bytes32 \_cUsername, bytes32 \_data)

Calling this function with relevant inputs shall modify the customer, along with removing the KYC request, if it exists. It also removes the votes casted, by assigning the variables upvotes and downvotes as 0, as well as refreshing the mapping banksVotedForCustomer. For instance, function is called with below arguments, shall remove the KYC request as well as the data regarding votes casted (upvotes/downvotes/ banksVotedForCustomer), corresponding to customer 1.

0x637573746f6d65725f3100000000000000000000000000000000000000000000, 0x637573746f6d65725f315f6d6f6469666965645f646174610000000000000000

**Remove KYC Request**

Function: removeKycRequest (bytes32 \_cUsername)

Calling this function shall remove the KYC request corresponding to a customer. Note any further voting actions will be limited hereon. For instance, calling the function with username 0x637573746f6d65725f3100000000000000000000000000000000000000000000, shall delete the KYC request corresponding to Customer 2.

**View Bank Details**

Function: viewBankDetails (address \_bankAddr)

This function returns the bank details. It can be called with any of the bank addresses mentioned above. The call shall fail if the bankAddr is invalid.

**Remove Bank**

Function: removeBank (address \_bankAddr)

This function shall remove the bank with given address.

**Report Bank**

Function: reportBank (address \_bankAddr)

This function can be accessed by one bank to report complaints against another bank. This also determines the voting status of the corrupt bank, once complaint is reported. If the number of complaints exceed 1/3rd of the total number of banks, then its voting rights are removed, until reverted by the admin.

**Get Complaints Against Bank**

Function: getBankComplaints (address \_bankAddr)

This function returns the number of complaints registered against a bank.

**Modify Bank’s Voting Status**

Function: modifyBankIsAllowedToVote (address \_bankAddr, bool \_allowed)   
  
This function is limited to the administrator. It overrides the voting status of a bank, until re-determined on reporting further complaints. Admin can also mark the status as false, regardless of the business logic (mentioned under heading **Report Bank**)