Instructions Phase-2

Step1: Compile and deploy Smart contract(admininterface.sol, kyc.sol ). After deploying contract, select any address from account(consider first account address as admin address).

Step2: Add banks using **AddBank** function. Parameters required to pass are

String \_Name e.g. “Neetu”

Address \_ethaddress e.g. select and copy any account address here(Bank1 address).

String \_regNumber e.g. “R1”

Similarly add total 5 banks by passing different parameters. Complaint reported is set as 0 and is allowed to vote is initially set as true within code for all banks.

Step3: View bank details using **viewBankdetail** function. Parameterrequired is

Address \_ethaddress

Step4: Add kycrequest using **addRequest function** . Parameter required is

String \_username e.g “preeti”

String \_customerdata e.g. “Aadhaar card no.”

Similarly add multiple customers kyc request. This can be done using any bank addresses.

Step5: Add customers using **addcustomer** function. Parameter required are

String \_usernameofthecustomer e.g. “preeti”

String \_customerdata e.g “Aadhaard card no.”

Initially kycstatus is set as false, downvotes is set as 0 and upvotes is st as 0 within code. This function can be executed by any bank account address.

Step6: view customer details using **viewcustomer** function. Parameter required is

String \_username of the customer

Any bank can view customer details.

Step 7: Any bank can upvote customer using **upvotecustomer** function. Parameter required is

String \_username of the customer

e.g upvote “preeti” by 3 bank addresses

Step 8: Any bank can downvote customer using **Downvotecustomer** function. Parameter required is

String \_username of the customer

e.g. down “preeti” by 1 bank addresses

Step 9: check upvoting and downvoting status using **viewcustomer** again.

Step10: Modify customer data using **modifycustomer**  function. Parameter required is

String \_usernameofthecustomer e.g. “preeti”

String \_newcustomerdata e.g “Aadhaard card no.”

Also status of initially set parameters can be changed based on conditions using this function. First condition is upvotes should be greater than downvotes. If this condition satisfies then check 2nd condition that downvotes should be less than 1/3 of the total no. of banks. Total banks in this contract is 5. If both satisfies then only this function changes status of initially set parameters.

e.g upvotes=3, downvotes= 1 then this function sets upvotes=0, downvotes=0 and kycstatus=true.

Step 11: View modified customer details using **viewcustomer** function.

Step 12: Kyc request can be removed using **removeRequest** function. Parameter required is

String \_usernameofthecustomer e.g. “preeti”

Step 13: A bank can report against any bank (who upvoted incorrect customer data) using function **reportBank.** Parameter required is

Address \_ethaddress e.g “Account address of Bank1”

This function sets the complaints reported parameter.

Step 14: Using **getBankcomplaints** function , fetch complaints reported against any bank. Parameter required is

Address \_ethaddress e.g. ethAddress is of bank1 account address that sets complaint reported against bank 1 as 1 and no any other bank reported complaint against this bank.

Step 15: Using **modifybankisallowedtovote** function, admin can modify is allowed to vote parameter from admin account address. Parameter required is

Address \_ethaddress

Status will change only if complaint reported by more than 1/3 of total number of banks.

e.g. As complaint reported was set by one bank as 1. So status of is allowed to vote will remain same i.e. true.This mean bank is valid and not corrupted.

Step 16: In case bank is found corrupted then then admin can execute function **removeBank** function to remove bank from kyc contract. Parameter required is

Address \_ethaddress (bank address).