**Smart Contract Flow**

* Once the contract is deployed, the address that deploy the contract will become Admin.
* Now admin can add the banks using method addBank() which accepts bank name, bank’s registration number and bank’s ethAddress.
* Once the banks are created now bank collects the information for the KYC from the customer.
* The information given by the customer includes username and customer data, which is the hash of the link for the customer data. This username is unique for each customer.
* A bank creates the request for submission using method addRequest(), which is stored in the smart contract.
* A bank then verifies the KYC data, which is then add the customer using addCustomer() method to the customer list.
* Other banks can get customer information from the customer list using viewCustomer().
* Other banks can also upvote/downvote customer data to demonstrate its authenticity using upVoteCustomer() and downVoteCustomer methods. If the number of upvotes exceeds the number of downvotes, then the kycStatus of the corresponding customer is set to true. If a customer is downvoted by one-third of the banks on the network, then their kycStatus is changed to false, even if the number of upvotes is greater than the number of downvotes
* A bank can also report another bank using reportBank() method and if it finds that the other bank is corrupt and is verifying false customers. Such corrupt banks will then be banned from upvoting/downvoting further. If complaints against a particular bank are received from more than one-third of the total. banks on the network, then that bank will be banned (i.e., isAllowedToVote will be set to false for that corrupt bank.)