Proof of Concept to Implement Distributed Ledger Technology for Anti-Money Laundering and Know Your Customer for the European Region

MSc Research Project
MSc In FinTech

Manish Shaw
Student ID: X19125135

School of Computing
National College of Ireland

Supervisor: Victor Del Rosal

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Manish Shaw		
Student ID:	X19125135		
Programme:	MSc In FinTech	**Year:**	2019-2020
Module:	Research Project		
Supervisor:	Victor Del Rosal		
Submission Due Date:	17th August 2020.		
Project Title:	"Proof of Concept to implement Distributed Ledger Technology for Anti-Money Laundering and Know Your Customer for the European Region"		
Word Count:	6287	**Page Count:**	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Manish Shaw

Date: 17th August 2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Proof of Concept to Implement Distributed Ledger Technology for Anti-Money Laundering and Know Your Customer for the European Region

Manish Shaw

X19125135

Abstract

There has been exponential growth in the use of digital technology around the globe in many businesses, due to the rapid technological advancement. The FinTech innovation like Machine learning, Artificial Intelligence and Distributed Ledger Technology (DLT) are revolutionizing the way business are being done today. One such industry that has been greatly influenced is the financial sector, with introduction of cryptocurrency as an instrument for buying and selling goods. This paper has objectives on implementation of DLT technology in a form of Proof of Concept for carrying out Anti-Money Laundering and Know Your Customer operation for the financial institution. As this from the first line of defence for the institution and the first point of user experience making it critical for the organisation. In the solution that user will be provide with control of the information and the organisation will be able to more transparent and auditable in line to the various financial regulation.

1 Introduction

For many decades the KYC and AML operations have been executed manually or have just been ignored by various financial institutions. It's only in the recent time that various governing bodies around the world have laid standards based on its demography. There is still no global regulation for the KYC and AML. In the European region there are few directives that define the AML and countering the financing of Terrorism (CFT) regulation. These operations can be carried out only if there are systems that are effective and efficient in keeping a systematic record of all such operations within an organisation. As in the old system where most of the work was done manual, it contained several loopholes leaving room for exploitation. There are individuals that are able to identify and take advantage of them to beat the system, so that they can infuse illicit funds into the financial system. Leading various organisations to end up paying huge penalties and subjected to various sanctions being imposed on them.

Distributed ledger technology (DLT) could be implemented here as it contains the potential for secure distribution of information among the stakeholders. DLT holds the ability to perform activities on a real time basis which help in gaining clarity on the operation and eases the complexity associated with processes like supervision by providing transparency. Blockchain technology which is increasingly talked about for its implementation on bitcoin is one of the various applications of distributed technology. It focuses on a peer to peer network with various forms of consensus mechanism and use of cryptography for hashing the communication (Nakamoto, 2008). It has been adopted in various industries like health care, supply chain, etc.

The research question is “Can DLT be used in AML and KYC to maintain the integrity of customer data and provide accountability & audibility within the customer onboarding operations for Financial Institutions?”.

The objective of this research is to provide a smart technology solution that can be implemented without compromising on the control and security of the data. This is intended to be achieved by implementing a pure decentralized distributed ledger system mentioned the components include:

1. Building a decentralized Application Programming Interface (API) using blockchain technology.
2. Providing control to the user over the data share on the blockchain.
3. Using Interplanetary File System (IPFS) to store data to maintain privacy of the customers.
4. Using smart contracts as a proof of concept to interact with the blockchain.

There are various works that have tried to implement blockchain in to build a tamper proof and secure solution. Where have tried to enhance transparency in the KYC process, where the customer is mapped to their information by them using public identifiers (Yadav & Chandak, 2019). They have also used various algorithms and symmetric keys to compress and encrypt the data shared by various customers (Sundareswaran, 2020).

2 Related Work

In this section will discuss on the works that have been done by researchers in field of Know Your Customer and Anti-money Laundering to counter illicit financial transaction and promoting terrorism. The various measure that are implemented by the financial institutions and government organizations by utilizing Distributed Ledger Technology to tackle the problem. Whereas countries like Switzerland (PLC, 2019) have issued strict guideline under the Financial Market Supervisory Authority- Anti- Money Laundering Ordinance (FINMA-AMLO) to control payment done using blockchain to create a level paying ground for the traditional business and the new innovative business.

Eric (Esoimeme, 2020) has suggested that the Nigerian legal department that they needs to stress on new implementing approaches for banking activity that have been applied in the developed nations. Goes on to suggesting the implementation of modern technologies like Distributer Ledger Technology, that it have been able reduce the cost and increase transparency in the KYC and AML processes. He is confident that on using this technology the Nigerian financial ecosystem will be able to reap its benefits.

(Lee, 2017) In her research has suggested that Blockchain technology will considerably improve efficiency and flexibility within the financial infrastructure. As this will reduce operations cost, enable cross-border payments, increase financial inclusions and require less budget of monitoring and compliance. As this will encourage government bodies to promote start-up business to increase business and boost the economy of the country. It is crucial that Financial Inclusion and AML activities need to be in balanced, there are many countries that do not press for identity management and ignore them, where as other have it as a business critical before building any business relationships (Esoimeme, 2020). Suggests that the governing bodies need to be proactive and regulated banks from allowing customer to hold multiple accounts and implementing blockchain as this technology will helps enforcing this with the banking network.

KYC is the crucial here because for this the current system is mostly physical that retains the risk of human error and exploitation. Nikolaos Kapsoulis and his colleagues proposed incorporating smart contracts and blockchain because it holds data on a centralized network that any approved staff such as banks would use to identify the individual (N. Kapsoulis, 2020). They hypothesize that KYC cycle will play a crucial role in financial protection as well as against other risks, and that a good KYC network will be critical. Jose and Ross claim there are two big advantages of use DLT for KYC testing (José Parra Moyano, 2017). The initial approach is the expense savings of the KYC method as a whole and the other is the removal of repetitive activities. In fact, it improves machine operation productivity even with several financial entities utilizing the network concurrently.

Another way is in depth understanding of the relation between Anti-Money Laundering and Digital Money which is the primary method for illegal money transfer. (Naheem, 2019) says that the traditional financial institution do not have the appropriate technologies to regulate digital currency, so they lack the expertise in this space. There need to update the knowledge on the technologies like Blockchain that are being adopted by new players. Also suggest that the only way to enhance the AML regulations and technology knowledge is by implementing the new technology into the existing system, which will provide level paying ground from them to get back into the game.

The approach here would be to keep records of digital footprint and Perri and Angela agree that these transactions can be tracked (Perri Reynolds, 2017). Blockchain has been marketed as an encrypted currency for a long time, but an in-depth study of transaction records will expose corrupt individuals in the scheme, the direct money for fraud. But this needs the current protection policies which is impractical to control the increasing online network so there is a need for tailored legislation to restrict these activities.

In their study, Karry Lai have spoken about how Singaporean banks use DLT as a method instead of tackling the problem of money laundering. Only because the government funded the banks that introduced these initiatives was this made possible. He further iterated that it was the government's guarantee to banks that allowed banks to confidently embrace DLT technology into their transaction. Present initiative often enables regulatory bodies to have a narrow image of the payments, but DLT may encourage them to get a broader vision when examining small aspects of a given transaction objectively (Lai, 2018).

Blockchain is now a big player in the financial world and Dancho, Bulgaria argues that the transformative existence of Blockchain will enable financial institutions to render their current systems more effective. In addition to increasing the time factor, the expense of the activities will often be greatly decreased along with other beneficial results. Its critical to `know the various positives and negatives of Distributed Ledger Technology specially in relation to legislative regulation. (Petrov, 2020)

The greatest danger found by Randy Priem in his study here is that all data is available in the program and is easily usable for all network users causing a privacy concern. Another problem is inconsistent DLTs, when the numbers of organisations are seeking to build their own specific networks that undermine the greater intent of this technology to support the larger economic ecosystem (Priem, 2020).

The details of all the transaction that are been recorded onto the blockchain in a chronological order as ledger. The ledger can be accessed with necessary permission, which will help regulators to verify the credibility of the individual involved in the transaction. In turn this will provide information that can be used to accelerate tracking down criminal activity and save cost of search at the same time (Esoimeme, 2020).

Ahmed (A. Alketbi, 2018) in his research talk about how that United Arab Emirates plan to implement blockchain into all the government activities, based on the used case for the early adopters like the UK, Singapore, Estonia, etc. In the paper they gone to focus of the key pointers that lay the foundation for the implementation, such as availability, integrity, accountability, authenticity, identity management and smart contract management. As the government's focus is reduce the involvement of human and central body's involvement, as this will lead to manipulation of information and malfunction of the system. As they intent to cash-out on the immutability characteristic of blockchain that does not permit and alteration to the transaction taken place to maintain transparency and accountability.

(Valkanov, 2019) In the research he stresses about the implementation of blockchain to facilitate secure share of information within the financial sector via permission to access the information of individual. As most of the banks are currently forced to maintain data of flagged individual and organisation and do not have any provision to share them among the counterparties. As this will help to expand the information on the entity, with trusted and quality data, that can be shared on real time bases. Which in turn help to provide a pooled resource for holding such information to reduce the cost of storage, verification and validation of data. Also goes on to take various used cases that have been able to implement DLT technology in their daily operations.

Mathieu and team (Mathieu Chanson, 2019) spoke about developing an IOT system based on four specify design requirement for implementing Blockchain. They were, one a tamper proof data processing, exchanging and creating, as building such a system is difficult. The second was the capability of handling high volume of data as IOT sensors generate huge amount of data. Third was to provide privacy while exchanging of data between the devices and is able to secure the communication. And the final was that the solution should be financially feasible to implement into the existing system.

Matthias, (Mettler, 2016) in the paper speaks of providing benefits to the healthcare patients by proposing a smart healthcare management system. Where he goes on to explain how blockchain will help chronic disease and elderly patients to provide a timely treatment by souring information for various healthcare stakeholder without disruption. Proposes to empower patients by proving them the control over information shared on the blockchain network.

3 Research Methodology

The most important activity in AML and KYC is the screening procedure which is an intense and tedious job of approving or rejecting users. The intention here is to create a blockchain database that contains AML and KYC verified users, that can be accessed by various trusted parties. The upgraded version will be an automated application making use of smart contracts, which saves time and minimizes the risk. The framework for this research is based on the Extreme Programming (XP) which is a known Agile Methodology, consisting of plan, design, code and test for the solution (Apoorva Singh, 2017). The proposed mechanism is further broken down based on the process for AML and KYC is illustrated below in the figure 1.

3.1 User Registration:

Users will have to undergo a registration process where individuals will have to register so that the KYC screening team can perform the background check on them. This will involve providing a standardized form that is mandatory to be filled out by the user. On completion of the form and clicking of the submit button the request will be sent to the KYC team for screening.

3.2 Screening team:

The KYC and AML team will perform the background screening on the user once their registration is complete by the user. At this stage there will be an intensive check on the user's background check. The team will investigate various angel and minor details of

Even minor errors could lead to various violations of regulations, which could lead to heavy penalties and sanctions for the organization. If the user is genuine, then the request will be approved by the team with a defined time period of its validity. No organization provides an infinite validity of approval to users and must be renewed at regular intervals. This would trigger the next step where the user will be assigned a smart contract to interact with the blockchain.

3.3 Smart Contract:

Once the request has been approved by the screening team, a smart contract is created to interact with the blockchain. As this requires information that will be used to authenticate the user's background, it would contain information like the user ID, the expiry date of their approval, etc. This will reduce the need for repeating the screening process again as it validates the request for further use of the information

3.4 IPFS:

Interplanetary File System is a decentralized system that works on the function peer to peer network for maintaining and sharing of data. This system provides a unique identity to the files which is known as cryptographic hashing. This reduces duplication from the network which helps in reducing the space needed to hold data.

3.5 Blockchain:

Blockchain is a decentralized distributed ledger system that stores information in various blocks of the network. It is a secure technology as it makes use of cryptography to store the information to protect this data from any external threats. All the information provided by the smart contracts are stored here. The information in the blockchain can be used for authentication of users as all the information is immutable in the blocks and cannot be changed. Even the user can check who has authorization to access their information in the blockchain. In case the validity of the user expires (expiry date of users depends on the organization policy), then the screening team will again carry out the verification process to assess the risk for the user.

3.6 Institution:

This will be the financial institution that will ultimately be granted access by the user to their personal and confidential information. This will be used in various business activities that the institution will focus on providing to the user like trading activities, transferring of funds for goods and service.

3.7 Execution:

For implementation of any technology or solution one must identify various steps that will be involved, which need to be combined to each other in a chronological order. On doing so, the process can be executed in a systematic way to achieve the desired results.

There is also a need for standardized guidelines on regulatory policies that would influence its implementation. As, there are various standards for AML and KYC validation that vary with the change in the region of the organization.

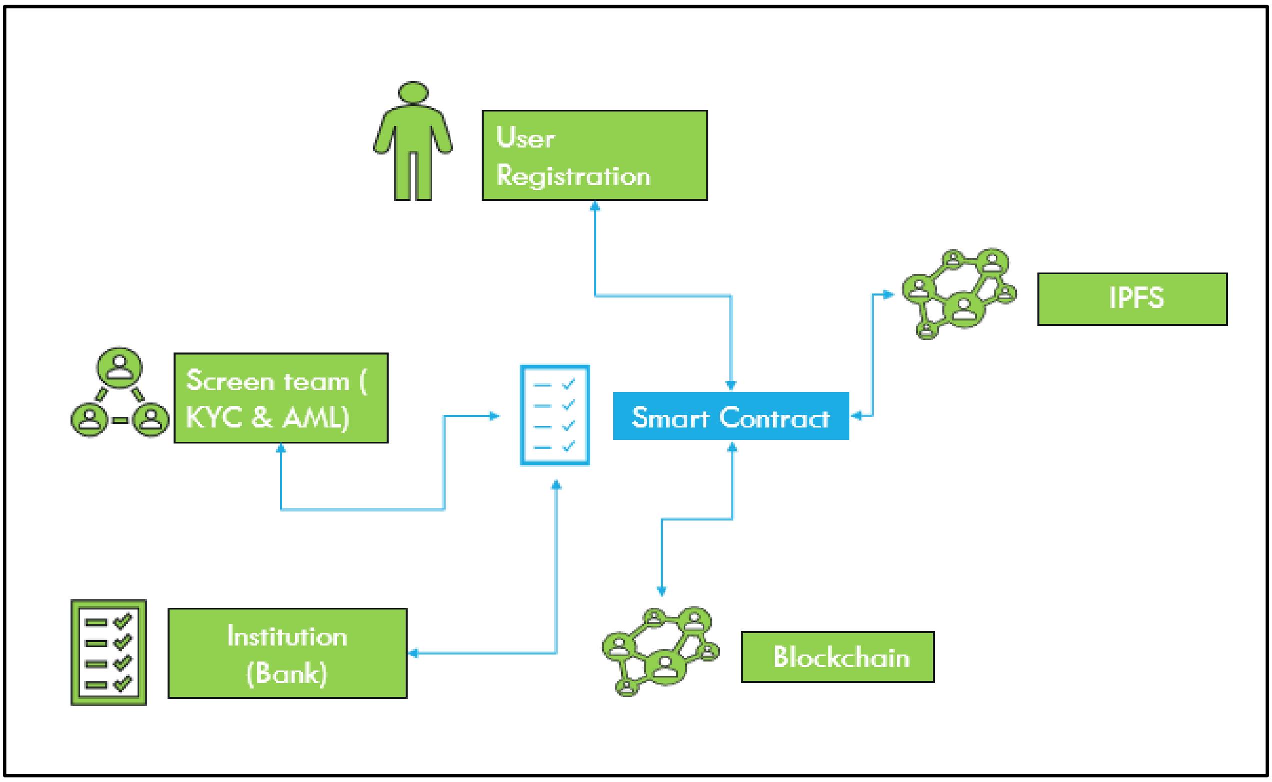


Figure 1: The visual flow chart of API solution.

4 Design Specification

In this section the brief of the application will be laid out, which will explain the various stages of the application's working. The distributed Application (DAPP) is divided into three parts, the backend, the frontend and the network connection that integrates the backend with the front end of the application. It is important that all the three are integrated to each other for the DAPP to work properly.

The Design of the Application has been done in such a way that it can be applied and customized to meet the business requirements of the business partners. This will be in line with the various financial regulations within the European region. Where the cost of implementation will be funded by the various business partners and not the users. As they will be provided with this service for no charges.

4.1 Back-End Application

This section is the backbone of the application which laid the foundation for the solution that is intended via this research the Proof of concept in the form of a smart contract which holds the operations logic. Smart contract is like a physical contract that contains the terms and conditions for doing business or for using goods and services in a digital form. It is called smart because it does not require human presence and can take decisions on its own depending on the logic that is used to build it. The Smart contract is the proof of concept that drives the application decision and actions. It is built using Solidity programming language, which is a high-level language that is object oriented mostly used to write smart contracts on multiple blockchain platforms like Ethereum, etc. This section is not visible to the user of the application, apart from the owners of the smart contract.

4.2 Front-End Application

In this section the end-users of the DAPP are able to see and use the various functionality that the owner of the application wants to share with the users. In this research the front-end will contain the form that has a predefined mandatory set of questions, the users and the banks need to fill. The information will be stored and passed down to the screening team as they are to be used for performing various transactions. This interface allows registered users to interact with the smart contract to fetch their information from the blockchain and check the statue of the entire transaction. This has been built by using HTML and JavaScript using React Web framework. HTML and React Web Framework are commonly used to build the front-end of a website where various features can be built using them which is appealing to the users. As they would not like to see clusters of java codes or scripts which require high technical know-how to understand and execute them. They help to provide a simple and easy to understand interface for the user.

4.3 Network Connection

It is important to establish a proper connection between the back end and the front-end, so that the application can provide an efficient and seamless experience for the various parties accessing the application. In this research the Truffle framework has been used to build the distributed application (DAPP). Truffle is a development framework used by blockchain developers to test and deploy the smart contract into the Ethereum Blockchain. It helps to maintain connection and interact with the various components like the smart contract, the blockchain and the various dependencies. It provides a powerful interactive console that can be used to access contracts and various commands in the framework. Ganache was used as the local Ethereum Blockchain during which it provided the project with ten accounts containing test Ethereum that can be used during the deployment and interaction within the node in the blockchain. As interaction with the actual

Ethereum blockchain requires real ether which is expensive and is not needed in the prototype development stage. MetaMask was used as the digital wallet as it provides a browser extension that maintains the test ether and is used while interacting with the smart contract. They combine to provide a complete package for the connection to the network for the application.

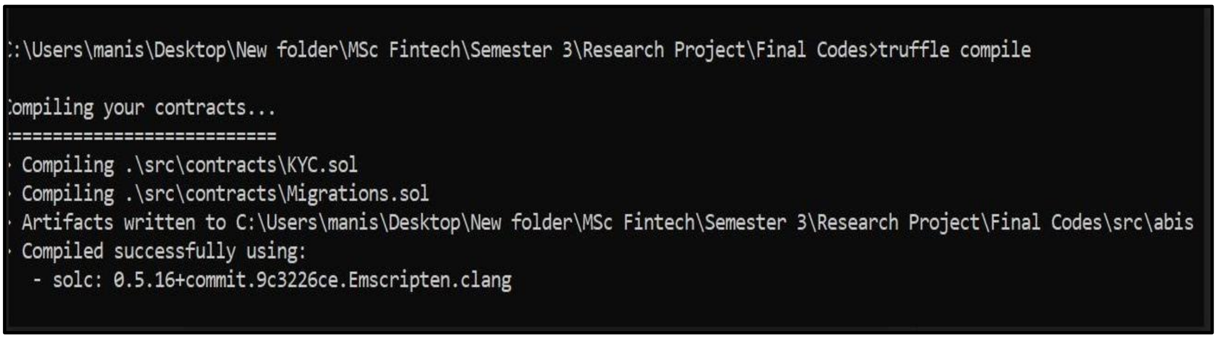
The above components form the three pillars of the distributed application that are seamlessly connected with each other. This will help to provide the right platform to solve the question and fulfil the objectives for the research.

5 Implementation

As the application is a prototype it is built using a local blockchain RPC, with multiple dependencies that must be downloaded and installed before being able to run the application. It is divided into five primary stages; each one of them are designed in such a way that the result is in line with the problem this project intends to solve. The Truffle framework will configure the various dependencies that are predefined in the configuration file to execute the components of the application. The output that is obtained after the implementation is of a controlled form that is connected to the blockchain to store and fetch information. Below are a few key aspects that describe the implementation to provide the desired outcome.

5.1 Compiling of the Smart Contract

The Smart contract used in this application is built using Solidity programming language, which is an object oriented statically typed language, forming the backbone of the DAPP. So, it is important that the source code must be compiled before it can be deployed onto the blockchain, the version of the compiler is predefined. This will be done by commanding Truffle suit to compile the application.

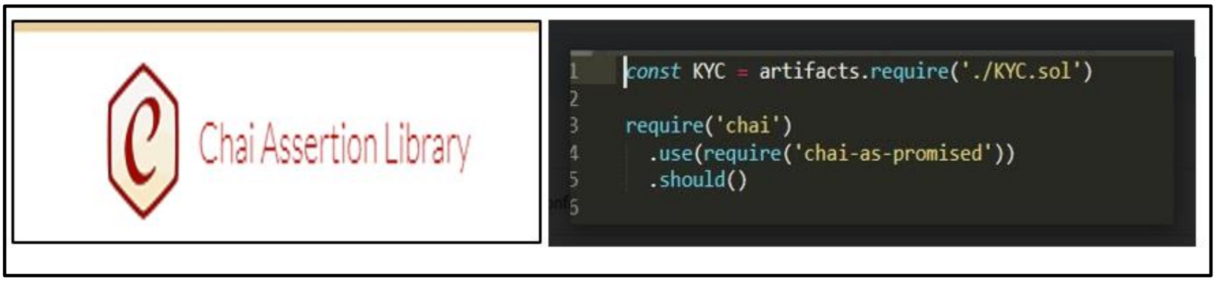


```
:\Users\manis\Desktop\New folder\MSc Fintech\Semester 3\Research Project\Final Codes>truffle compile

ompiling your contracts...
==========================
· Compiling .\src\contracts\KYC.sol
· Compiling .\src\contracts\Migrations.sol
· Artifacts written to C:\Users\manis\Desktop\New folder\MSc Fintech\Semester 3\Research Project\Final Codes\src\abis
· Compiled successfully using:
   - solc: 0.5.16+commit.9c3226ce.Emscripten.clang
```

Figure 2: Successfully Compiling the Smart Contract

5.2 Testing the Smart Contract

In Blockchain all the transaction is immutable and cannot be changed once deployed onto the blocks. It is important that one needs to run various tests to identify the various bugs and errors in the source code (Nakamoto, 2008). This application must pass the predefined test before being deployed on to the local blockchain. They are written using Chai Assertion Library (Groups, 2020) which is a JavaScript testing framework to test the various functionality of the DAPP. The testing is also done on the Remix IDE provided by the Ethereum to develop, test and deploy the smart contract on the multiple Blockchain Networks.



```
const KYC = artifacts.require('./KYC.sol')

require('chai')
  .use(require('chai-as-promised'))
  .should()
```

Figure 3: The Testing Framework Used for Smart Contract Deployment

5.3 Deploying the Smart Contract

Once the compiling and testing have been completed on the source code, the next process is deployment of the smart contract (POC) onto the blockchain. The migration function on the truffle suite provides the feature to deploy the smart contract to the local blockchain which in this case is Ganache the local RCP. At this stage the backend of the distributed application has been successfully set up and live for integration onto the frontend.

5.4 Deploying the User Interface

In the design section it is known that the user interface is built on the React Web Framework, which is a framework that allows developers to build an Application Programming Interface (API). The React framework used JavaScript for building the API, in this project its store and called from the Source directory in the components folder. It will be deployed by requesting node.js to run the react component on the command prompt and it will deploy the API on the localhost which is set to “localhost:3000”.

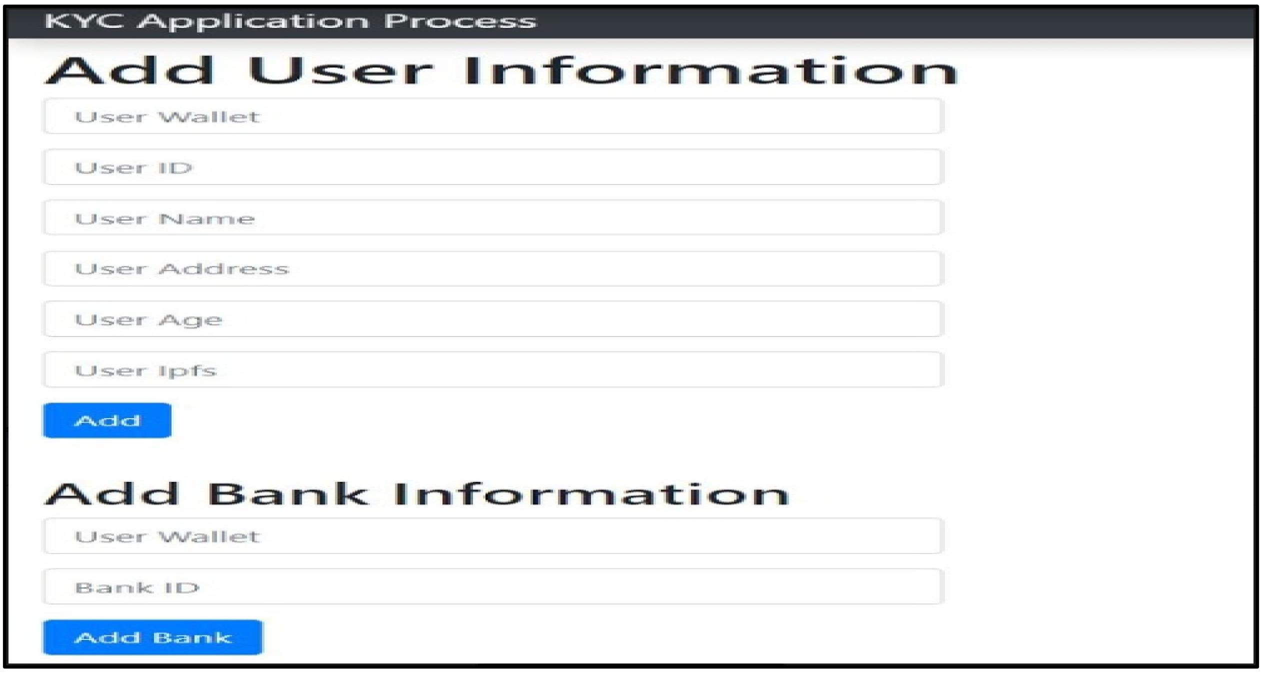


Figure 4: The Front-end Form on the API.

5.5 Performing the Various Functions

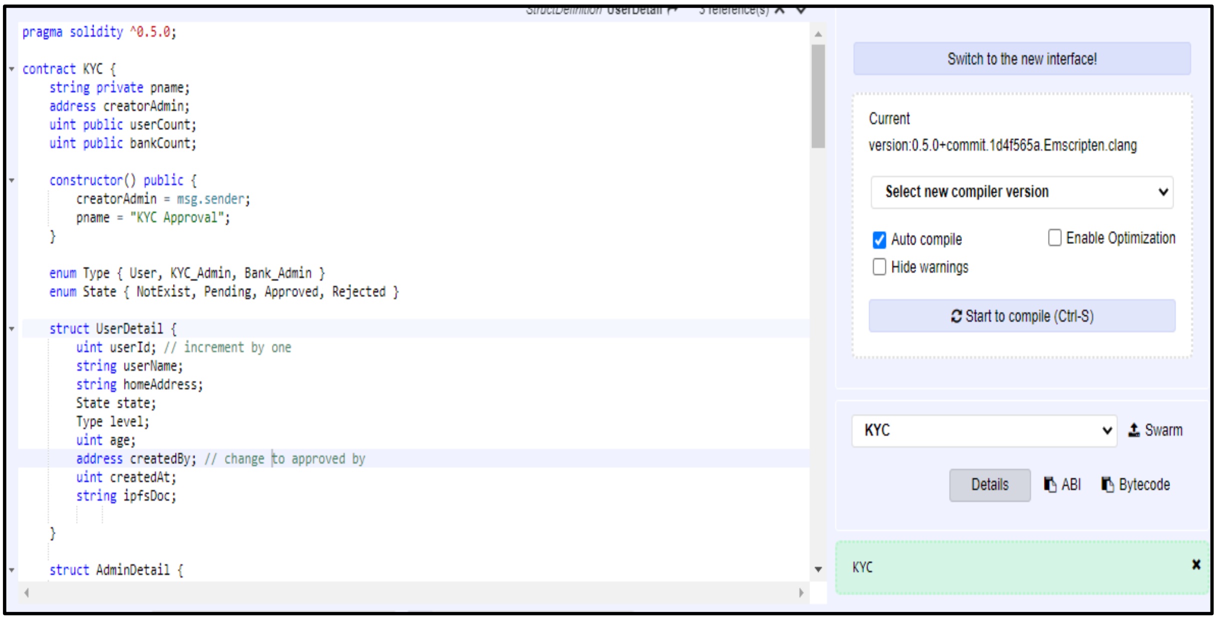
When the API is launched the form for the user and bank will be seen which will accept only individuals at a time be it the bank or the user. This API form will help to store the information on to the blockchain and be able to fetch the information of the registered user. Will not allow any outsider to access any information and will restrict duplicate entries.

6 Evaluation

The focus here is to test the practical implementation of the solution proposed in this research. The various components will be tested here, and the outcome will be discussed, if the solution has been able to address the issues. The test is divided into stages as per the chronological order of implementation. The Test is conducted using the Chai Assertion library (Groups, 2020) testing framework shown in figure and along with Remix IDE provided by Ethereum.

6.1 Compiling the Smart Contract

In this stage the test will be done to see if the smart contract is able to pass the compiling stage or there has been any error in the formulation of the source code of the contract. The version here for the compiler is 0.5.0 for the pragma solidity, so that it compiles the source code the way it was intended. It can be noticed that source code has been compiled correctly in the image file --. Also, on Remix IDE after compiling the colour of the contract has turned green indicating that the compilation was successful. Otherwise it would have thought errors indicated in red, with the error messages showing what is wrong in the source code.



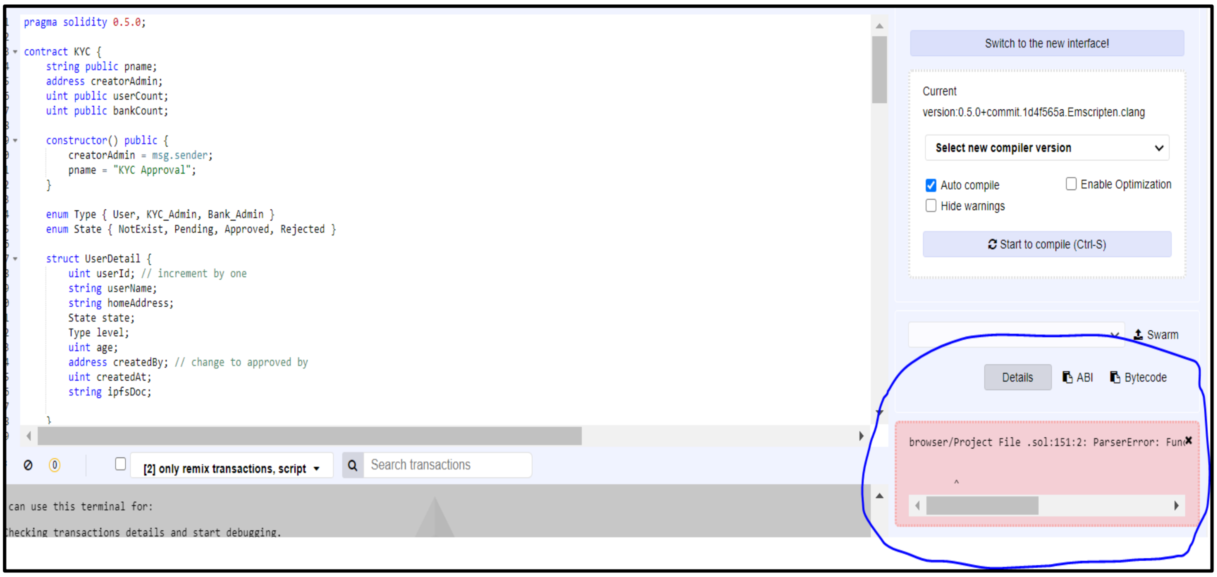
StructDefinition UserDetail
3 reference(s)
Switch to the new interface!
Select new compiler version
Start to compile (Ctrl-S)
KYC
Swarm
Details
ABI
Bytecode
KYC

```
pragma solidity ^0.5.0;

contract KYC {
    string private pname;
    address creatorAdmin;
    uint public userCount;
    uint public bankCount;

    constructor() public {
        creatorAdmin = msg.sender;
        pname = "KYC Approval";
    }

    enum Type { User, KYC_Admin, Bank_Admin }
    enum State { NotExist, Pending, Approved, Rejected }

    struct UserDetail {
        uint userId; // increment by one
        string userName;
        string homeAddress;
        State state;
        Type level;
        uint age;
        address createdBy; // change to approved by
        uint createdAt;
        string ipfsDoc;

    }

    struct AdminDetail {
```

Current
version:0.5.0+commit.1d4f565a.Emscripten.clang

☑ Auto compile
□ Enable Optimization
□ Hide warnings

Figure 5: Compiling of the Smart Contract using Remix IDE.



```
pragma solidity 0.5.0;

contract KYC {
    string public pname;
    address creatorAdmin;
    uint public userCount;
    uint public bankCount;

    constructor() public {
        creatorAdmin = msg.sender;
        pname = "KYC Approval";
    }

    enum Type { User, KYC_Admin, Bank_Admin }
    enum State { NotExist, Pending, Approved, Rejected }

    struct UserDetail {
        uint userId; // increment by one
        string userName;
        string homeAddress;
        State state;
        Type level;
        uint age;
        address createdBy; // change to approved by
        uint createdAt;
        string ipfsDoc;

    }
```

```
can use this terminal for:
hecking transactions details and start debugging.
```

Current
version:0.5.0+commit.1d4f565a.Emscripten.clang

☑ Auto compile ☐ Enable Optimization
☐ Hide warnings

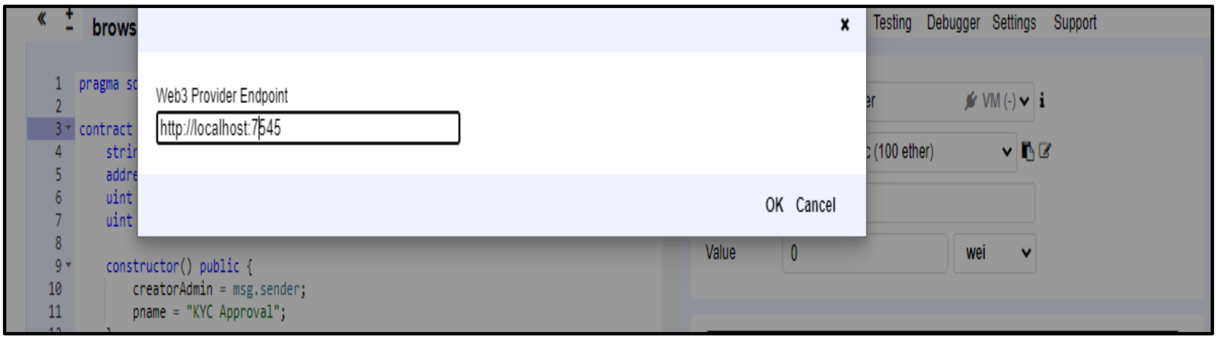
```
browser/Project File .sol:151:2: ParserError: Fun
        ^
```

Switch to the new interface!
Select new compiler version
Start to compile (Ctrl-S)
Swarm
Details
ABI
Bytecode
[2] only remix transactions, script
Search transactions

Figure 6: Error Message while compiling the Smart Contract on Remix IDE.

6.2 Deployment test for the Smart Contract

In this section the test will be done to check the deployment of the smart contract correctly onto the local Blockchain. On the Remix IDE, there are multiple fields that need to be filled up before deploying the smart contract to the local blockchain which is Ganache for this research. The environment of the IDE needs to change to the localhost, which Ganache at IP address 127.0.0.1:7585 where the network ID will be 5777. Once that is configured, the contact can be deployed on to the blockchain by clicking on the ‘Deploy’ button.

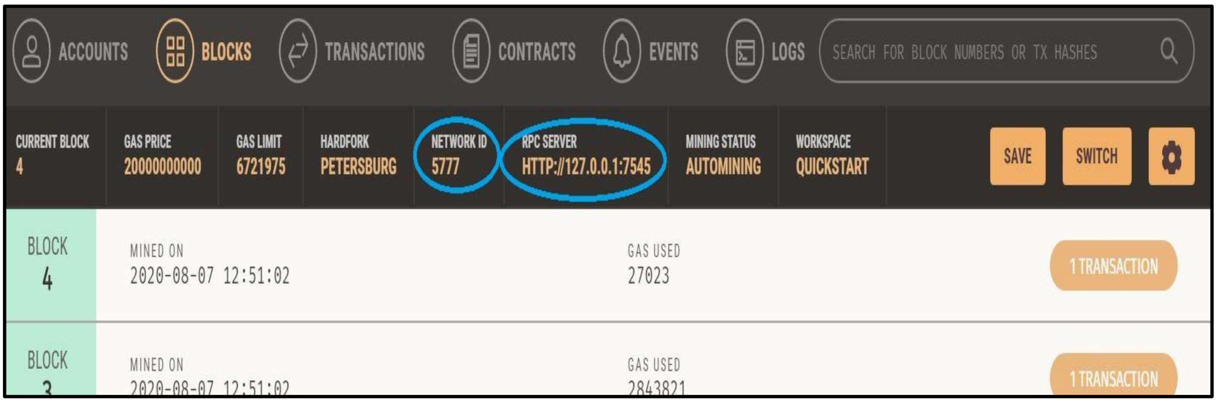


Web3 Provider Endpoint
http://localhost:7545

```
pragma so
contract
    stri
    addre
    uint
    uint

    constructor() public {
        creatorAdmin = msg.sender;
        pname = "KYC Approval";
```

Value 0 wei

Figure 7: Configuring of Local Blockchain Ganache (RPC) to Remix IDE.

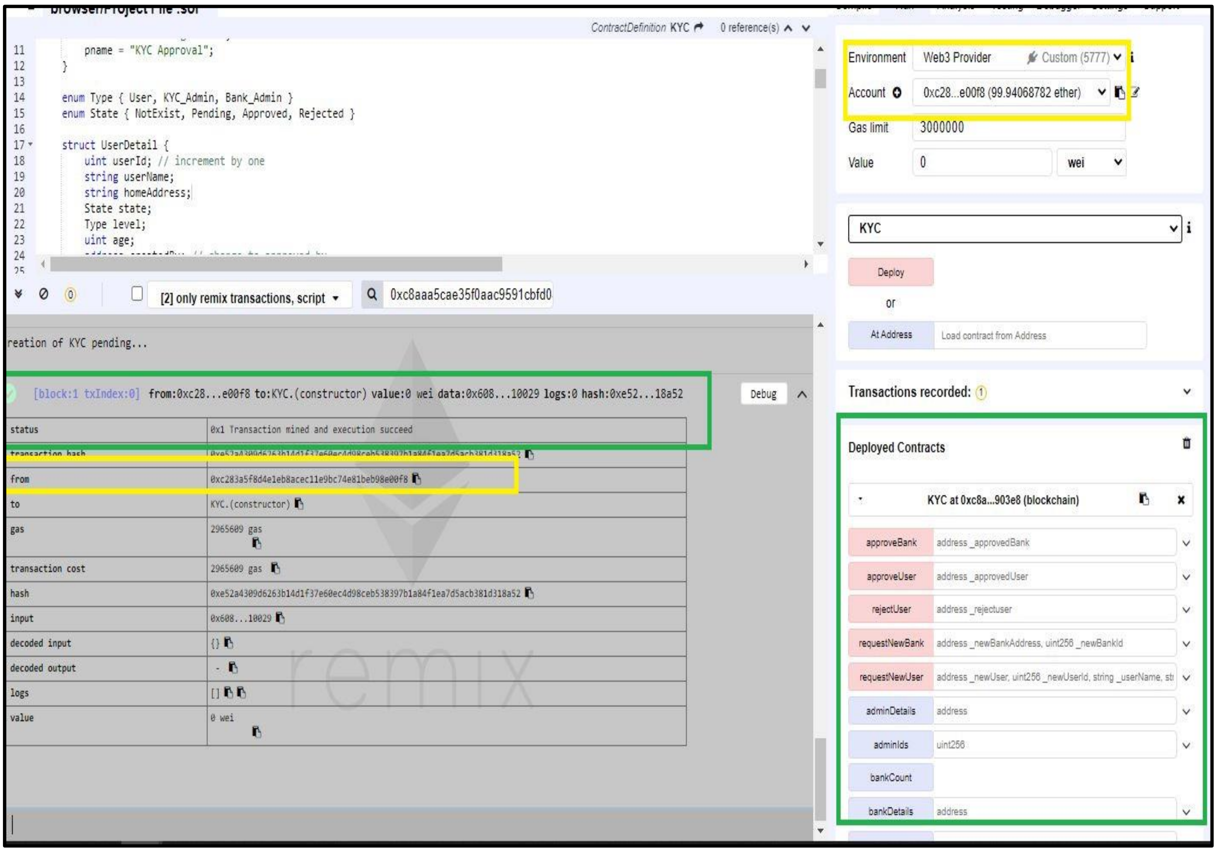


CURRENT BLOCK 4
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK PETERSBURG
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE QUICKSTART

BLOCK 4
MINED ON 2020-08-07 12:51:02
GAS USED 27023
1 TRANSACTION

Figure 8: Ganache (RPC) server Location and Network ID.

After deployment it was noticed that the smart contract was successfully deployed on the blockchain marked in green colour, in the figure below with a green tick symbolizing success in the execution. It can further be noticed that there are blocks that have been created on the ganache dashboard and the transaction cost being charged shown in the IDE console. The account that owns the contract is highlighted in yellow and the form for storing and fetching has started to appear on the run console marked in green on the left of indicating that the contract is successfully been deployed to the blockchain.



```
        pname = "KYC Approval";
    }

    enum Type { User, KYC_Admin, Bank_Admin }
    enum State { NotExist, Pending, Approved, Rejected }

    struct UserDetail {
        uint userId; // increment by one
        string userName;
        string homeAddress;
        State state;
        Type level;
        uint age;
```

```
reation of KYC pending...

[block:1 txIndex:0] from:0xc28...e00f8 to:KYC.(constructor) value:0 wei data:0x608...10029 logs:0 hash:0xe52...18a52
```

status	0x1 Transaction mined and execution succeed
from	0xc283a5f8d4e1eb8acec11e9bc74e81beb98e00f8
to	KYC.(constructor)
gas	2965609 gas
transaction cost	2965609 gas
hash	0xe52a4309d6263b14d1f37e60ec4d98ceb538397b1a84f1ea7d5acb381d318a52
input	0x608...10029
decoded input	{}
decoded output	-
logs	[]
value	0 wei

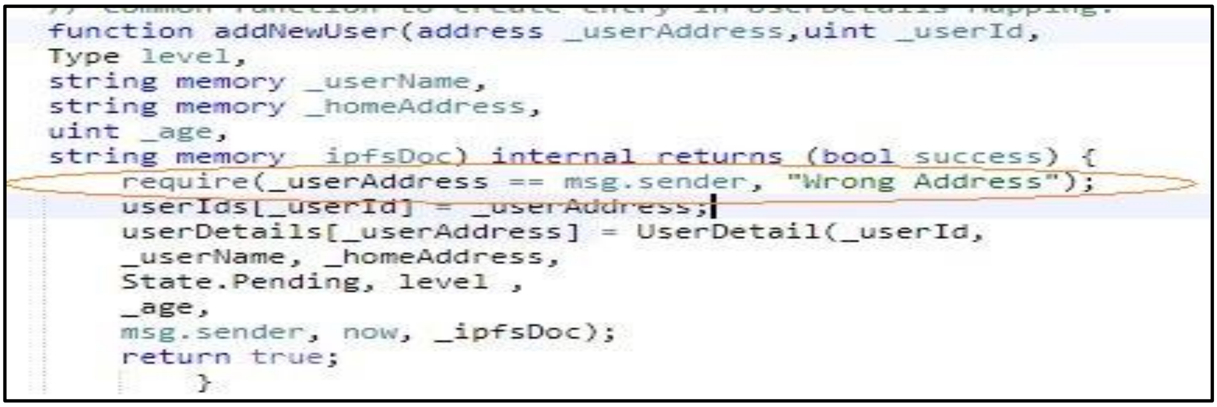
Environment: Web3 Provider — Custom (5777)
Account: 0xc28...e00f8 (99.94068782 ether)
Gas limit: 3000000
Value: 0 wei
KYC

approveBank	address _approvedBank
approveUser	address _approvedUser
rejectUser	address _rejectuser
requestNewBank	address _newBankAddress, uint256 _newBankId
requestNewUser	address _newUser, uint256 _newUserId, string _userName, st
adminDetails	address
adminIds	uint256
bankCount	
bankDetails	address

Figure 9: Deployment of the Smart contract onto Local Blockchain.

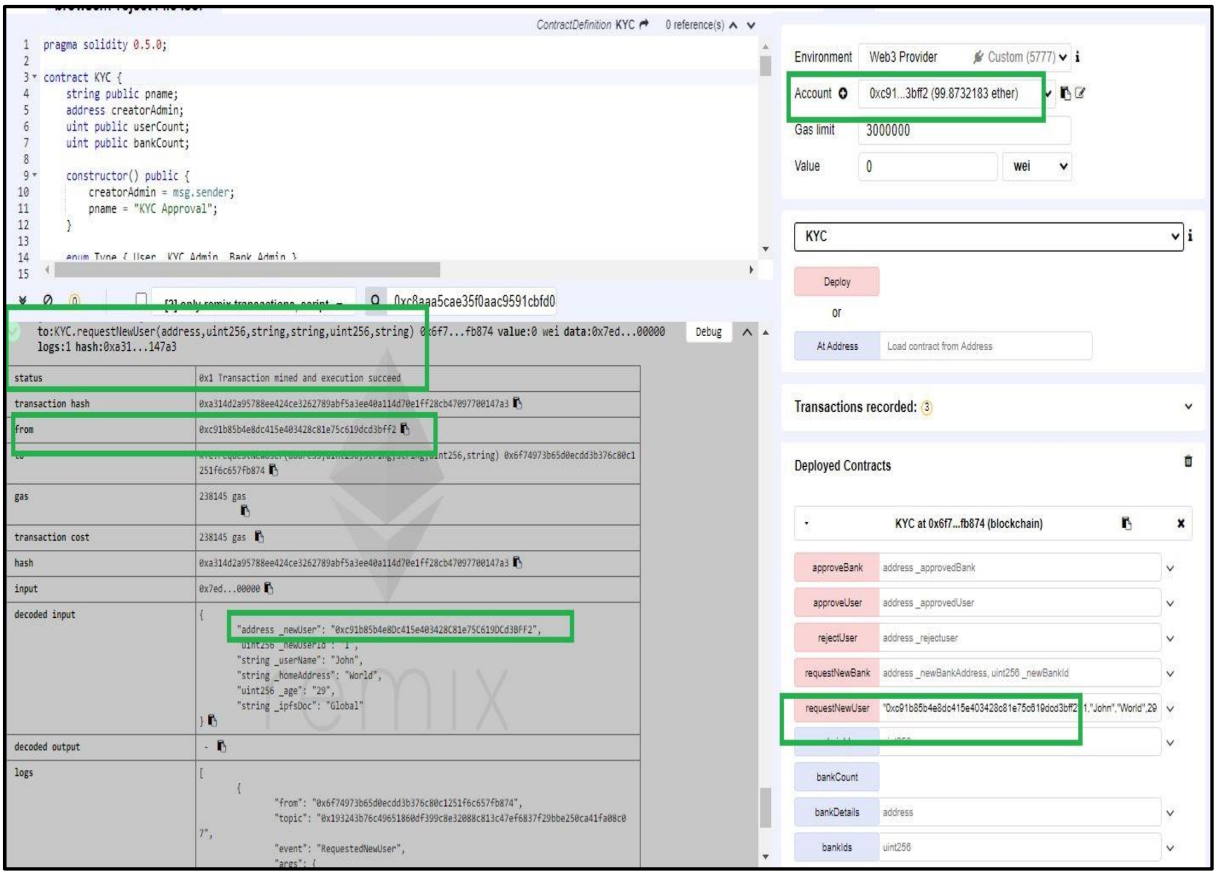
6.3 Test the control over sharing of user information

On successfully completing the deployment of the contract the form will become active for storing and fetching the information from the local blockchain. Here the intent of the test is to check if the user and the account getting registered are the same. The ‘required’ function will make sure that the send account details match to the account that is getting registered and not someone else adding the information shown in the figure below.



```
function addNewUser(address _userAddress,uint _userId,
Type level,
string memory _userName,
string memory _homeAddress,
uint _age,
string memory _ipfsDoc) internal returns (bool success) {
    require(_userAddress == msg.sender, "Wrong Address");
    userIds[_userId] = _userAddress;
    userDetails[_userAddress] = UserDetail(_userId,
    _userName, _homeAddress,
    State.Pending, level ,
    _age,
    msg.sender, now, _ipfsDoc);
    return true;
        }
```

Figure 10: The Function that Allows only User to register.



ContractDefinition KYC
0 reference(s)

```
pragma solidity 0.5.0;

contract KYC {
    string public pname;
    address creatorAdmin;
    uint public userCount;
    uint public bankCount;

    constructor() public {
        creatorAdmin = msg.sender;
        pname = "KYC Approval";
    }

    enum Type { User, KYC Admin, Bank Admin }
```

[2] only remix transactions, script
0xc8aaa5cae35f0aac9591cbfd0
Debug

```
to:KYC.requestNewUser(address,uint256,string,string,uint256,string) 0x6f7...fb874 value:0 wei data:0x7ed...00000
logs:1 hash:0xa31...147a3

status              0x1 Transaction mined and execution succeed
transaction hash    0xa314d2a95788ee424ce3262789abf5a3ee40a114d70e1ff28cb47097700147a3
from                0xc91b85b4e8dc415e403428c81e75c619dcd3bff2
to                  KYC.requestNewUser(address,uint256,string,string,uint256,string) 0x6f74973b65d0ecdd3b376c80c1
                    251f6c657fb874
gas                 238145 gas
transaction cost    238145 gas
hash                0xa314d2a95788ee424ce3262789abf5a3ee40a114d70e1ff28cb47097700147a3
input               0x7ed...00000
decoded input       {
                            "address _newUser": "0xc91b85b4e8Dc415e403428C81e75C619DCd3BFF2",
                            "uint256 _newUserId": "1",
                            "string _userName": "John",
                            "string _homeAddress": "World",
                            "uint256 _age": "29",
                            "string _ipfsDoc": "Global"
                    }
decoded output       -
logs                [
                            {
                                    "from": "0x6f74973b65d0ecdd3b376c80c1251f6c657fb874",
                                    "topic": "0x193243b76c49651860df399c8e32088c813c47ef6837f29bbe250ca41fa08c0
                    7",
                                    "event": "RequestedNewUser",
                                    "args": {
```

Environment: Web3 Provider — Custom (5777)
Account: 0xc91...3bff2 (99.8732183 ether)
Gas limit: 3000000
Value: 0 wei

KYC

Transactions recorded: 3

Deployed Contracts

KYC at 0x6f7...fb874 (blockchain)

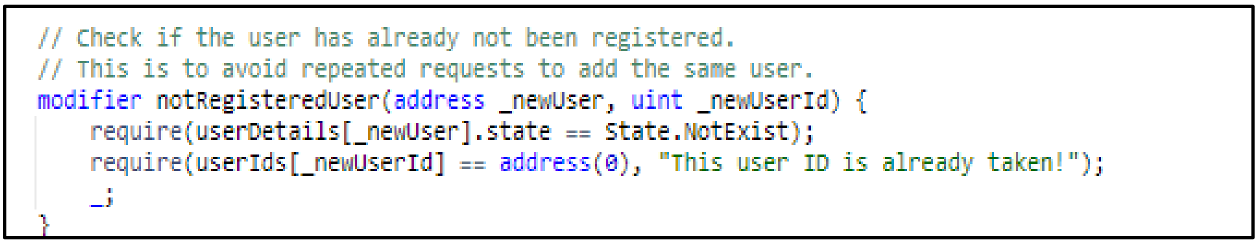
requestNewUser: "0xc91b85b4e8dc415e403428c81e75c619dcd3bff2",1,"John","World",29

Figure 11: Successfully Registering User Information onto the Blockchain.

It was noticed that the smart contract has accepted the user only when the requesting account match to the one provided for registration. Even the owner of the smart contract was not permitted to register on behalf of the user or the bank.

6.4 Testing for duplicate entries of Users and Banks

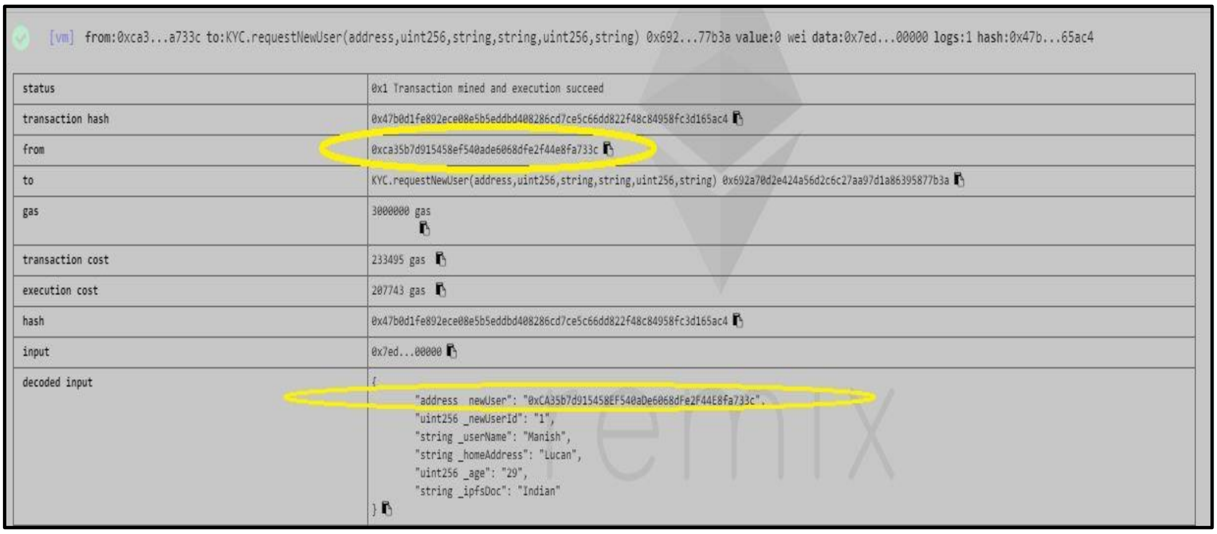
It is important that the smart contract restricts creation of a duplicate account for the user or the bank onto the blockchain. The DAPP has been designed to this as well, the below function on the smart contract enables the feature.



```
// Check if the user has already not been registered.
// This is to avoid repeated requests to add the same user.
modifier notRegisteredUser(address _newUser, uint _newUserId) {
    require(userDetails[_newUser].state == State.NotExist);
    require(userIds[_newUserId] == address(0), "This user ID is already taken!");
    _;
}
```

Figure 12:Function to Restrict double Entry.

On testing this the feature performed the task it was designed to perform, as it would not allow the transaction to be performed when the individual tries to register twice.



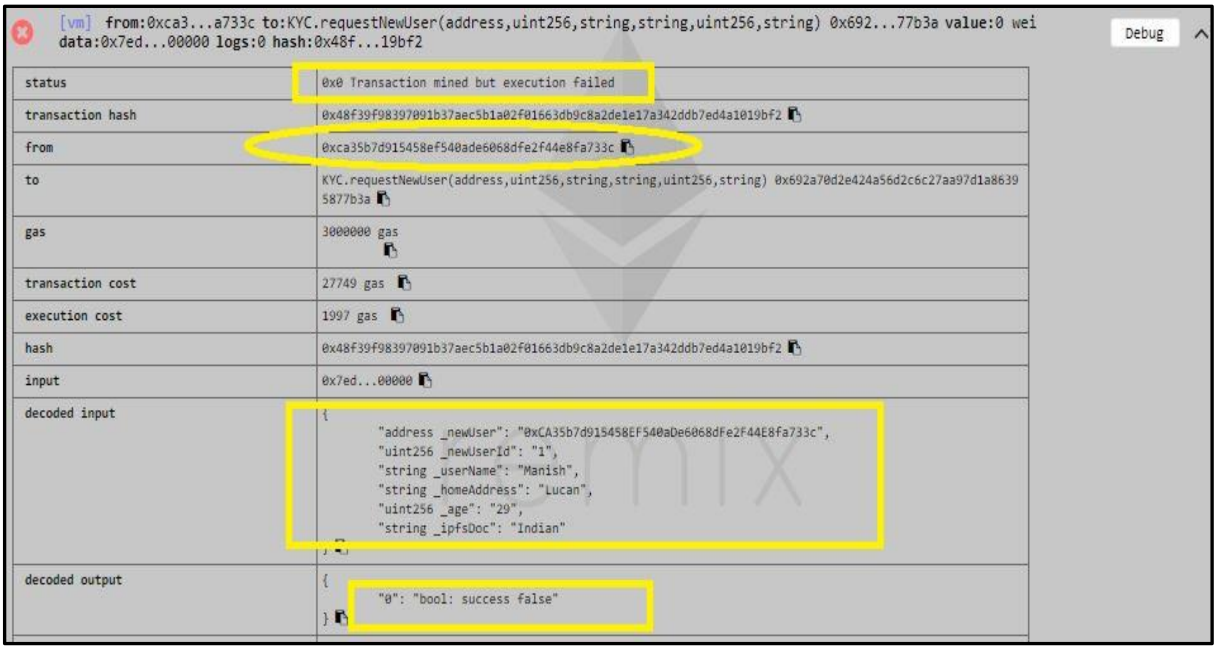
```
[vm] from:0xca3...a733c to:KYC.requestNewUser(address,uint256,string,string,uint256,string) 0x692...77b3a value:0 wei data:0x7ed...00000 logs:1 hash:0x47b...65ac4
```

status	0x1 Transaction mined and execution succeed
transaction hash	0x47b0d1fe892ece08e5b5eddbd408286cd7ce5c66dd822f48c84958fc3d165ac4
from	0xca35b7d915458ef540ade6068dfe2f44e8fa733c
to	KYC.requestNewUser(address,uint256,string,string,uint256,string) 0x692a70d2e424a56d2c6c27aa97d1a86395877b3a
gas	3000000 gas
transaction cost	233495 gas
execution cost	207743 gas
hash	0x47b0d1fe892ece08e5b5eddbd408286cd7ce5c66dd822f48c84958fc3d165ac4
input	0x7ed...00000

decoded input	{
	"address _newUser": "0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c",
	"uint256 _newUserId": "1",
	"string _userName": "Manish",
	"string _homeAddress": "Lucan",
	"uint256 _age": "29",
	"string _ipfsDoc": "Indian"
	}

Figure 13: Evidence to prove the requested and the registered account are same.

As it can be noticed the user registering for the first time is able to perform the task successfully shown on the status section on the above figure. Also, it can be noticed that the account which performed the translation is the same as the User on the decoded input address. When the attempt was made to add the same account again with the same User Address or the User ID the system rejected the request, it can be seen in the figure below.



```
[vm] from:0xca3...a733c to:KYC.requestNewUser(address,uint256,string,string,uint256,string) 0x692...77b3a value:0 wei
data:0x7ed...00000 logs:0 hash:0x48f...19bf2
```

status	0x0 Transaction mined but execution failed
transaction hash	0x48f39f98397091b37aec5b1a02f01663db9c8a2de1e17a342ddb7ed4a1019bf2
from	0xca35b7d915458ef540ade6068dfe2f44e8fa733c
to	KYC.requestNewUser(address,uint256,string,string,uint256,string) 0x692a70d2e424a56d2c6c27aa97d1a86395877b3a
gas	3000000 gas
transaction cost	27749 gas
execution cost	1997 gas
hash	0x48f39f98397091b37aec5b1a02f01663db9c8a2de1e17a342ddb7ed4a1019bf2
input	0x7ed...00000

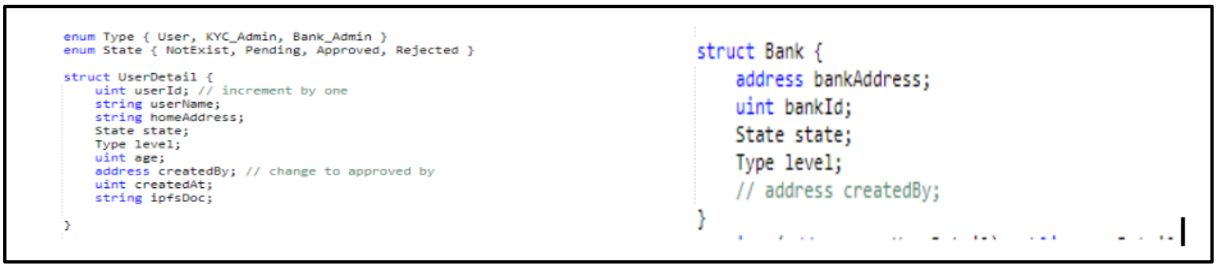
decoded input	{
	"address _newUser": "0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c",
	"uint256 _newUserId": "1",
	"string _userName": "Manish",
	"string _homeAddress": "Lucan",
	"uint256 _age": "29",
	"string _ipfsDoc": "Indian"
	}

decoded output	{
	"0": "bool: success false"
	}

Figure 14: Rejection of Duplicate Request for Registration.

6.5 Tracking the KYC Screening Status

For the process to complete it is important that the user is able to track the status of the application, the user is able to access the information at any time after the registration is completed. There were multiple runs where the user and the bank were able to track the status of the requesting system. The status is predefined to ‘pending’ when that request is registered for adding the accounts, which changes only after approval by the KYC team. The category is set based on the type of request, in the case of users are ‘User’ and banks are ‘Bank_Admin’. They are done by the function called ‘enum’ ; they are an array that starts from zero to store information.

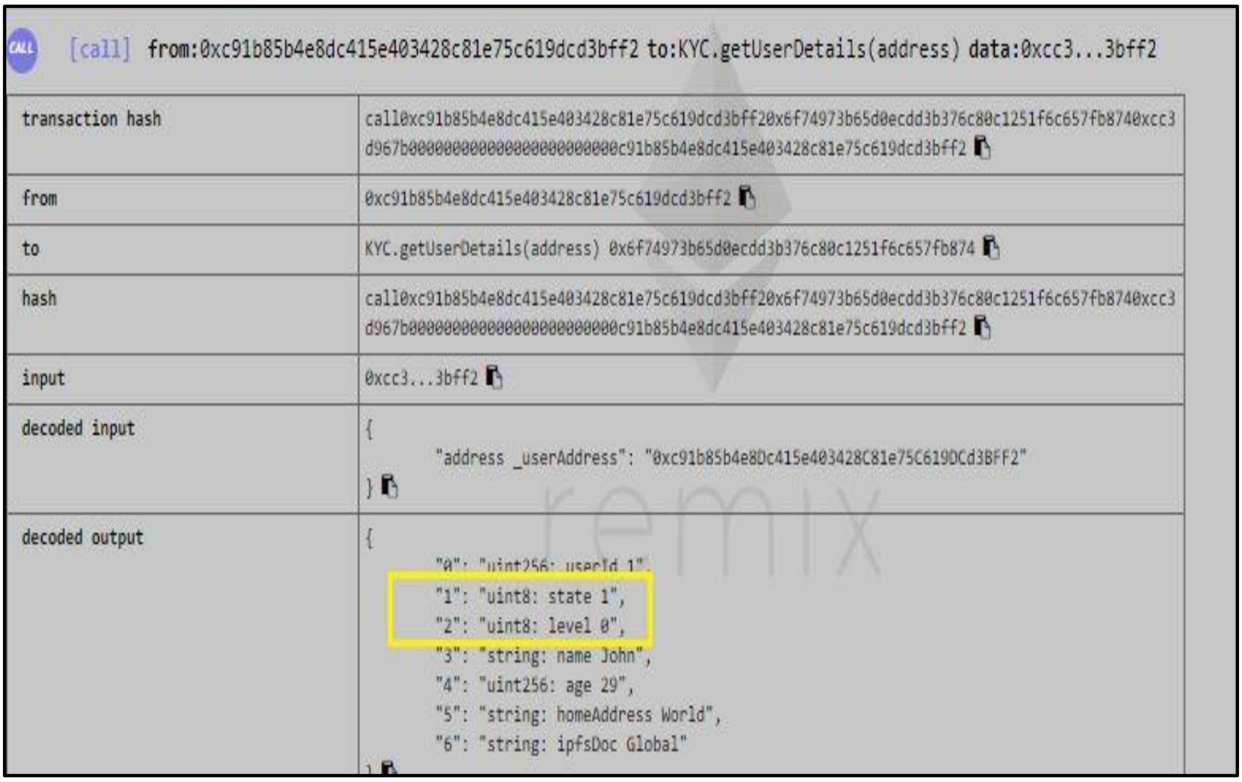


```
enum Type { User, KYC_Admin, Bank_Admin }
enum State { NotExist, Pending, Approved, Rejected }

struct UserDetail {
    uint userId; // increment by one
    string userName;
    string homeAddress;
    State state;
    Type level;
    uint age;
    address createdBy; // change to approved by
    uint createdAt;
    string ipfsDoc;
}
```

```
struct Bank {
    address bankAddress;
    uint bankId;
    State state;
    Type level;
    // address createdBy;
}
```

Figure 15: ‘enum’ used for Defining the type and status of the accounts based on the request.

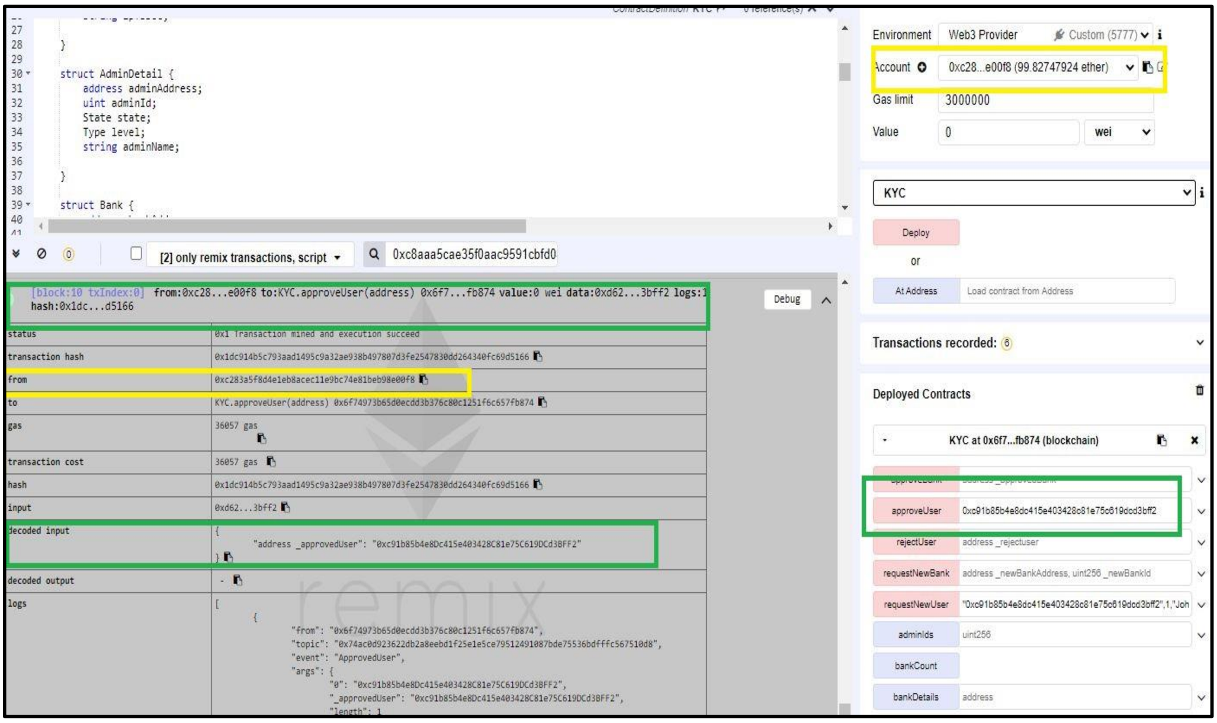
It can be noticed in the figure below that approval has not been provided where the user ‘state’ is showing “1” means that status is still pending it can change post approval.



```
[call] from:0xc91b85b4e8dc415e403428c81e75c619dcd3bff2 to:KYC.getUserDetails(address) data:0xcc3...3bff2

transaction hash   call0xc91b85b4e8dc415e403428c81e75c619dcd3bff20x6f74973b65d0ecdd3b376c80c1251f6c657fb8740xcc3
                   d967b000000000000000000000000c91b85b4e8dc415e403428c81e75c619dcd3bff2
from               0xc91b85b4e8dc415e403428c81e75c619dcd3bff2
to                 KYC.getUserDetails(address) 0x6f74973b65d0ecdd3b376c80c1251f6c657fb874
hash               call0xc91b85b4e8dc415e403428c81e75c619dcd3bff20x6f74973b65d0ecdd3b376c80c1251f6c657fb8740xcc3
                   d967b000000000000000000000000c91b85b4e8dc415e403428c81e75c619dcd3bff2
input              0xcc3...3bff2
decoded input      {
                         "address _userAddress": "0xc91b85b4e8Dc415e403428C81e75C619DCd3BFF2"
                   }
decoded output     {
                         "0": "uint256: userId 1",
                         "1": "uint8: state 1",
                         "2": "uint8: level 0",
                         "3": "string: name John",
                         "4": "uint256: age 29",
                         "5": "string: homeAddress World",
                         "6": "string: ipfsDoc Global"
                   }
```

Figure 16: Status of the User after registration.

Once the request is sent to the KYC team for approval, they are the only authority that can approve the request as shown in the test run below as they are not permitted to make any other changes onto the account.



```
    }

    struct AdminDetail {
        address adminAddress;
        uint adminId;
        State state;
        Type level;
        string adminName;

    }

    struct Bank {
```

[2] only remix transactions, script
0xc8aaa5cae35f0aac9591cbfd0
Debug

```
[block:10 txIndex:0] from:0xc28...e00f8 to:KYC.approveUser(address) 0x6f7...fb874 value:0 wei data:0xd62...3bff2 logs:1
hash:0x1dc...d5166

status             0x1 Transaction mined and execution succeed
transaction hash   0x1dc914b5c793aad1495c9a32ae938b497807d3fe2547830dd264340fc69d5166
from               0xc283a5f8d4e1eb8acec11e9bc74e81beb98e00f8
to                 KYC.approveUser(address) 0x6f74973b65d0ecdd3b376c80c1251f6c657fb874
gas                36057 gas
transaction cost   36057 gas
hash               0x1dc914b5c793aad1495c9a32ae938b497807d3fe2547830dd264340fc69d5166
input              0xd62...3bff2
decoded input      {
                         "address _approvedUser": "0xc91b85b4e8Dc415e403428C81e75C619DCd3BFF2"
                   }
decoded output     -
logs               [
                         {
                               "from": "0x6f74973b65d0ecdd3b376c80c1251f6c657fb874",
                               "topic": "0x74ac0d923622db2a8eebd1f25e1e5ce79512491087bde75536bdfffc567510d8",
                               "event": "ApprovedUser",
                               "args": {
                                     "0": "0xc91b85b4e8Dc415e403428C81e75C619DCd3BFF2",
                                     "_approvedUser": "0xc91b85b4e8Dc415e403428C81e75C619DCd3BFF2",
                                     "length": 1
```

Environment: Web3 Provider — Custom (5777)

Account: 0xc28...e00f8 (99.82747924 ether)

Gas limit: 3000000

Value: 0 wei

KYC

Transactions recorded: 6

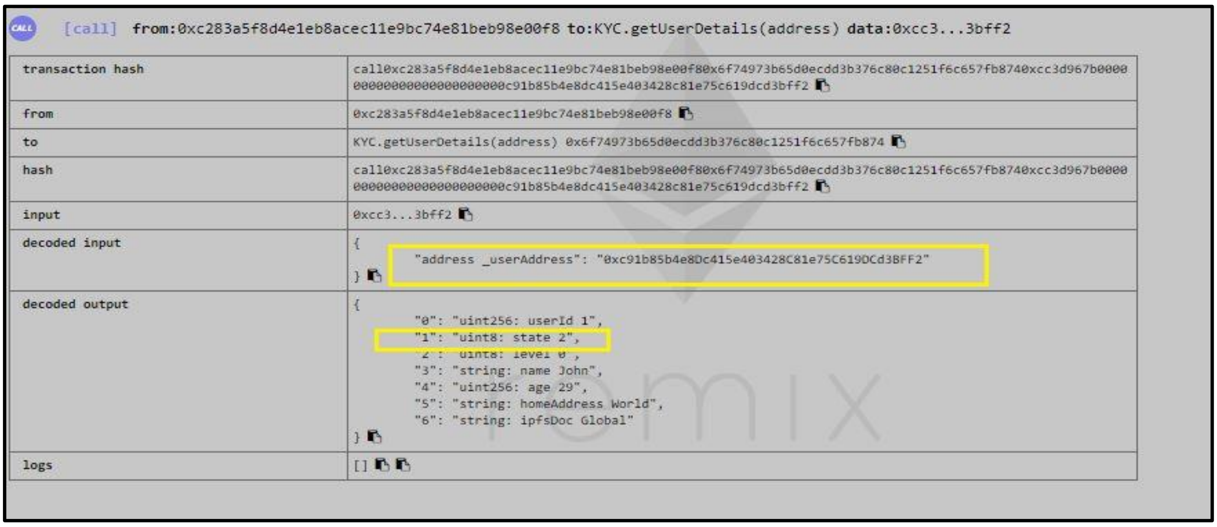
Deployed Contracts

KYC at 0x6f7...fb874 (blockchain)

approveBank	address _approvedBank
approveUser	0xc91b85b4e8dc415e403428c81e75c619dcd3bff2
rejectUser	address _rejectuser
requestNewBank	address _newBankAddress, uint256 _newBankId
requestNewUser	"0xc91b85b4e8dc415e403428c81e75c619dcd3bff2",1,"Joh
adminIds	uint256
bankCount	
bankDetails	address

Figure 17: Approving of the User by the KYC Team/Owners of the Contract.

On approval the owners of the account are able to fetch the information from the block and will be able to view the status on the account has changed illustrated in the figure below.



[call] from:0xc283a5f8d4e1eb8acec11e9bc74e81beb98e00f8 to:KYC.getUserDetails(address) data:0xcc3...3bff2

transaction hash	call0xc283a5f8d4e1eb8acec11e9bc74e81beb98e00f80x6f74973b65d0ecdd3b376c80c1251f6c657fb8740xcc3d967b0000 00000000000000000000c91b85b4e8dc415e403428c81e75c619dcd3bff2
from	0xc283a5f8d4e1eb8acec11e9bc74e81beb98e00f8
to	KYC.getUserDetails(address) 0x6f74973b65d0ecdd3b376c80c1251f6c657fb874
hash	call0xc283a5f8d4e1eb8acec11e9bc74e81beb98e00f80x6f74973b65d0ecdd3b376c80c1251f6c657fb8740xcc3d967b0000 00000000000000000000c91b85b4e8dc415e403428c81e75c619dcd3bff2
input	0xcc3...3bff2

decoded input	{
	"address _userAddress": "0xc91b85b4e8Dc415e403428C81e75C619DCd3BFF2"
	}

decoded output	{
	"0": "uint256: userId 1",
	"1": "uint8: state 2",
	"2": "uint8: level 0",
	"3": "string: name John",
	"4": "uint256: age 29",
	"5": "string: homeAddress World",
	"6": "string: ipfsDoc Global"
	}

logs	[]

Figure 18: Account Status after Approval.

6.6 Discussion

It is evident that DLT has been able to impact various businesses irrespective of the industry type and AML and KYC screening are the very first stage of customer onboarding. Being able to reduce errors, increase auditability and tamper-proof system for the organisation helps to minimize fraud and crime. This research has focused on building one, that can be implemented into the daily activities of a financial institute. As financial institutes are easy targets for money laundering and victims of financial crimes and fraud transactions. If they are able to control them at the onboarding stage it is unlikely that they would be victim of such forgery.

The prototype was designed keeping in mind the multiple objectives, like making the process completely digital and more customer centric. Where the customer is given the control of the information shared and cannot be tampered by the third-party. It can be noted that in the test conducted in the above section only the user is allowed to input the information and no one else, not even the owner or the Smart contracts. This will help to maintain the purity and integrity of the information. The users are able to track the information in the system on their own, when it gets updated immediately on completion of the KYC and AML screening activity.

The other aspect of the smart contract is being stored in a decentralized platform to provide continuous access to the information. Which was illustrated using Ganache as the Local RPC that was used for developing the blockchain solution for the test environment, where the smart contract was stored on multiple blocks within the blockchain. Provides evidence of a Proof of Concept in form of Smart Contracts running on a platform built using a Distributed Ledger Technology.

The Smart contact is able to provide the feature to restrict the creation of duplicate entries for an account that has been registered on the system. This helps to reduce double spending on evaluating an existing user, in turn reducing the load on the system for maintaining and account processing.

It can be noticed that the prototype will be able to lay the foundation for a stronger system to prevent various illicit activities and provide a transparent system (Esoimeme, 2020). The evaluation has been able to highlight the various features that will be powerful tools for the organisation to control illegal activities and misuse of information. There is much more that can be done that is not covered in this research which will be in the future works of this research.

7 Conclusion and Future Work

The proof of concept is a smart contract built on the DLT platform called Blockchain. That interact within a peer to peer network the block making it tamperproof and has well-defined roles and responsibilities for the various parties involved in the system. The other objective was to provide control to the user/customer over their personal information without the involvement of a third person to maintain the integrity of the data. A financial institution has had a tough job to execute with the rapid advancement in the use of technology and tracking the whereabouts of its customers and sources of funds (Perri Reynolds, 2017)The proposed system will be able to solve multiple problems and lay a strong foundation by being able to increase transparency, accountability, and control over the AML and KYC operations. The function like 'requires' that only allows the user and the bank had the authority to provide their data and restrict anyone from altering the information is vital. Also, discourage users from creating duplicate accounts to miss lead and manipulate the screening operations carried out by the team. The application was able to provide evidence on what it performs the remaining of the artefacts that will be developed and implemented in future work.

The Future work will involve in building a fully functional IPFS system that will be a part of the solution which will involve storing of confidential files like passport, Identity cards, tax returns and balance sheets share with authorised personals only. These data will be further used to develop another Proof of Concept to assess the Financial Risk associated with the account in respect to Credit Risk Models (Kolodinsky, Sep2009)calculated by implementing Machine Learning, Forecasting and Prediction techniques (Jinwen Sun, 2019). As these are the next steps that are involved in the onboarding process for new users/ customers to build business relationship by performing financial transactions (Weytjens, et al., 2019).

8 References

A. Alketbi, Q. N. a. M. A. T., 2018. Blockchain for government services — Use cases, security benefits and challenges. *Learning and Technology Conference,* 1(15), pp. 112-119.

Apoorva Singh, D. P., 2017. Implementation of Requirement Engineering in Extreme Programing and SCRUM. *International Journal of Advanced Research in Computer Science,* 8(5), pp. 621-624.

Esoimeme, E. E., 2020. Balancing anti-money laundering measures and financial inclusion ,The example of the United Kingdom and Nigeria. *Journal of Money Laundering Control,* 23(1), pp. 64-79.

Esoimeme, E. E., 2020. Institutionalising the war against corruption: new approaches to assets tracing and recovery. *Journal of Financial Crime,* 27(1), pp. 217-230.

Jinwen Sun, K. X. C. L. W. Z. H. X., 2019. Exploiting intra-day patterns for market shock prediction: A machine learning approach. *Expert Systems With Applications,* Volume 127, pp. 272-281.

José Parra Moyano, O. R., 2017. KYC Optimization Using Distributed Ledger Technology. *Business & Information Systems Engineering,* Volume 59, pp. 411-423.

Kolodinsky, J. M., Sep2009. Objective measures as a predictor of late payments by high-risk borrowers.. *Erin. International Journal of Consumer Studies,* 33(5), pp. 591-595.

Lai, K., 2018. Singapore banks using DLT to tackle money Launderying. *International Financial Law Review ; London (Mar 28, 2018)..*

Lee, E., 2017. A Challenge to the New Paradigm of Financial Technology, Regulatory Technology and Anti-Money Laundering Law. *journal of Business Law,* Issue 6, pp. 473-498.

Mathieu Chanson, A. B. D. B. E. F. F. W., 2019. Blockchain for the IoT: Privacy-Preserving Protection of Sensor Data. *Journal of the Association for Information Systems,* 20(9), pp. 1271-1307.

Mettler, M., 2016. Blockchain Technology in Healthcare,The Revolution Starts Here. *International Conference on e-Health Networking,* Volume 18.

N. Kapsoulis, A. P. G. P. A. M. A. L. a. T. V., 2020. Know your customer (KYC) implementation with smart contracts on a privacy-oriented decentralized architecture. *Future Internet,* 12(41).

Naheem, M. A., 2019. Exploring the links between AML, digital currencies and blockchain technology. *Journal of Money Laundering,* 22(3), pp. 515-526.

Nakamoto, S., 2008. Bitcoin: a peer-to-peer electronic cash system..

Perri Reynolds, A. S. I., 2017. Tracking digital footprints: anonymity within the bitcoin system. *Journal of Money Laundering Control,* 20(2), pp. 172-189.

Petrov, D., 2020. Blockchain Ecosystem in the Financial Services Industry. 8(1).

PLC, E. I. I., 2019. Switzerland: Payments on blockchain; stringent Swiss AML standards for regulated financial intermediaries. *International Financial Law Review.*

Priem, R., 2020. Distributed ledger technology for securities clearing and settlement: benefits, risks, and regulatory implications. *Financial Innovation,* 6(11).

Sundareswaran, N., 2020. Optimised KYC Blockchain System. *International Conference on Innovative Trends in Information Technology,* pp. 1-6.

Valkanov, N., 2019. Smart Compliance or How New Technologies Change Customer Identification Mechanisms in Banking. *ELECTRONIC JOURNAL "ECONOMICS AND COMPUTER SCIENCE" ,* Issue 2.

Weytjens, H., Lohmann, E. & Kleinsteuber, M., 2019. Cash flow prediction: MLP and LSTM compared to ARIMA and Prophet. *Electronic Commerce Research;,* 1(1), pp. 1-21.

William J. Gordon, C. C. d., 2018. Blockchain Technology for Healthcare: Facilitating the Transition to Patient-Driven Interoperability. *Computational and Structural Biotechnology Journal,* 16(1), pp. 224-230.

Yadav, P. & Chandak, R., 2019. Transforming the Know Your Customer (KYC) Process using Blockchain. *International Conference on Advances in Computing, Communication and Control (ICAC3),* pp. 1-5.