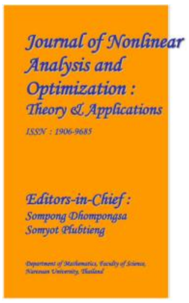
Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 15, Issue. 1, No.15 : 2024
ISSN : **1906-9685**


Journal of Nonlinear Analysis and Optimization : Theory & Applications
ISSN : 1906-9685
Editors-in-Chief :
Sompong Dhompongsa
Somyot Plubtieng

SECURE SHARING OF IDENTITY (KYC)

K. Rambabu [1], Bandaru DevakiNandan [2],

[1]**Assistant professor(HOD) , PG DEPT,** Dantuluri Narayana Raju College**, Bhimavaram, Andharapradesh**
Email:- kattarambabudnr@gmail.com
[2]**PG Student of MCA, Dantuluri** Narayana Raju College**, Bhimavaram, Andharapradesh**
Email:- bandarudevakinandan@gmail.com

ABSTRACT

In the contemporary era, approximately 70% of user data is readily available online, stored in centralized servers where the potential for misuse or alteration by owners or database managers poses a significant threat. Users often find themselves compelled to disclose sensitive information, including address, zip code, gender, and medical history, for identity verification or Know Your Customer (KYC) registrations. The lack of a direct mechanism for users to ascertain the security of their data on these servers raises concerns about privacy breaches and unauthorized use of personal information.

To mitigate the risks associated with the leakage of sensitive information, we propose the implementation of Blockchain-based KYC Sharing. Blockchain technology, renowned for its inherent support for data encryption, access control, and data immutability, presents a robust solution. Each record in the Blockchain is treated as a transaction or block, uniquely associated with a HASHCODE. During the addition of new records, Blockchain meticulously verifies the HASHCODE of all preceding records. If the data remains unaltered, the verification process is successful, rendering the Blockchain immutable. Access to data stored in the Blockchain is restricted to users with explicit permissions, ensuring secure sharing of records.

1 INTRODUCTION

In our increasingly digitized world, the proliferation of online platforms has resulted in a substantial volume of user data, particularly through identity verification processes like Know Your Customer (KYC). However, the conventional methods of storing and sharing identity information often fall short in terms of security, leaving users vulnerable to data breaches and misuse. The "Secure Sharing of Identity (KYC)" project aims to address these concerns by leveraging advanced technologies to establish a robust and trustworthy framework for managing and sharing sensitive identity information.

Literature Survey

1.Title :KYC Optimization by Blockchain Based Hyperledger Fabric Network

Author: Nazir Ullah; Kawther A. Al-Dhlan; Waleed Mugahed Al-Rahmi

Abstract:

In financial institutions, the traditional Know Your Customer (KYC) process is insecure, and costly. The adoption of disruptive technology is a necessary condition for the coming future of financial institutions. This study proposed a Hyperledger Fabric network for KYC optimization. The Performance of the proposed system was tested by using the Hyperledger Composer. The experiment results confirm that due to the strong and identity features of Hyperledger Fabric, the proposed system can speed up the KYC clearing transfers, challenge the inefficiencies that arise from duplicated conduct of similar tasks, secure data sharing, cost-effective, and ultimately bring transparency in the traditional KYC system.

3 IMPLEMENTATION STUDY

EXISTING SYSTEM:

Currently, the majority of identity data is stored in centralized systems, posing inherent risks of unauthorized access, manipulation, and potential breaches. Users, especially during KYC procedures, are often required to disclose sensitive information, including personal details and documentation, without adequate assurance of its security. This reliance on centralized servers lacking advanced security features exposes users to the ever-looming threat of identity theft, privacy invasion, and misuse.

Disadvantages:

- **Security Audits and Upgrades**: Conduct regular security audits and implement robust encryption, authentication, and access control mechanisms.
- **Investment in Technology**: Allocate resources for upgrading technologies and training personnel to manage secure KYC data sharing effectively.
- **Risk Management**: Develop a comprehensive risk management framework to identify and mitigate operational risks associated with KYC data handling.

Proposed System & alogirtham

To enhance the security and privacy of identity information, the proposed "Secure Sharing of Identity (KYC)" project introduces a state-of-the-art solution by incorporating blockchain technology. Blockchain, renowned for its immutable and decentralized nature, ensures data integrity, access control, and encryption. Each identity record becomes a secure transaction within the blockchain, associated with a unique identifier. The system employs smart contracts to facilitate secure data sharing, allowing access only to authorized parties while maintaining a transparent and traceable record of data interactions.

4.1 Advantages:

- **Strong Encryption**: Implement state-of-the-art encryption algorithms (e.g., AES-256) to ensure sensitive KYC data is securely stored and transmitted.
- **Flexible Integration**: Develop APIs with clear specifications and standards to facilitate seamless integration with third-party services, legacy systems, and external stakeholders.

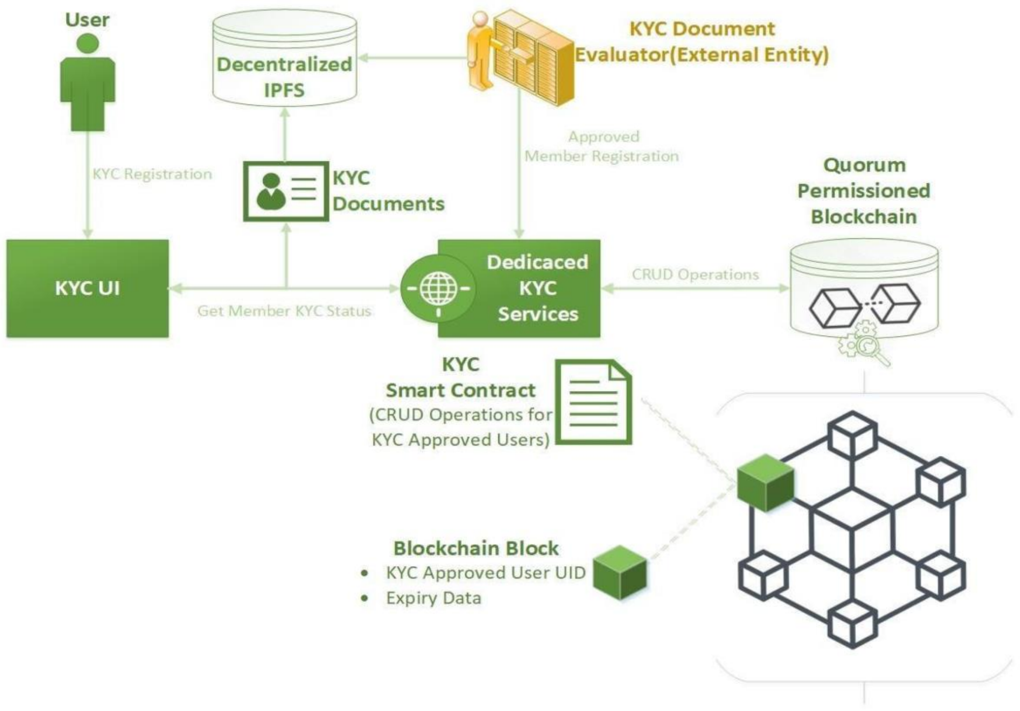


Fig:3.1 System Architecture

IMPLEMENTATION

Modules Used in Project :

Tensorflow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

Numpy : Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

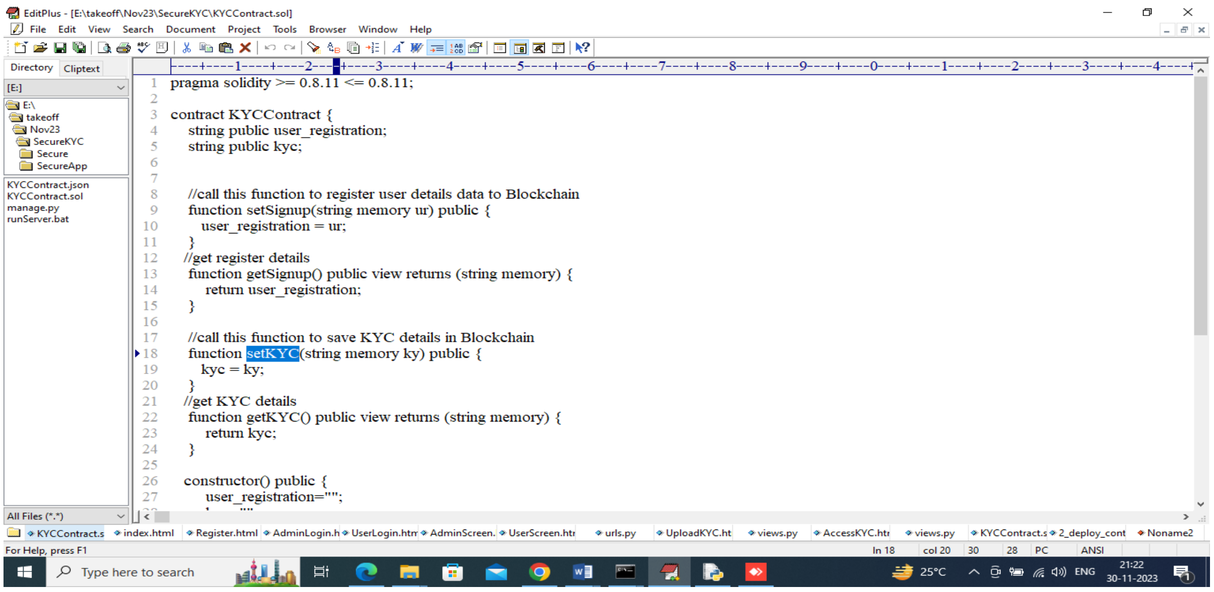
5 RESULTS AND DISCUSSION

Screenshots:

Now-a-days worldwide 70% user's data are available online and this data will be stored at single centralized servers whose owners or database managers may misuse or alter user's data and there is no direct way for the user to know about such misuse. Sometime for Identity proof or KYC registrations users may have to disclose sensitive information to such servers and leakage of such sensitive information may harm user's privacy. Sensitive information include address, zip code, gender, diseases etc.

To avoid such leakage of sensitive information we are employing Blockchain based KYC Sharing. Blockchain has inbuilt support for data encryptions, data access control and immutability (data once stored can be altered in any manner) of stored data. Blockchain store each record as transaction or block and associate each block with unique HASHCODE and while storing fresh records Blockchain will verify HASHCODE of all previous records and if data unchanged then it will result into successful verification and this verification will make Blockchain immutable. Data stored in Blockchain can be access by only those users who has permissions and can be consider as Secure Sharing of Records. In propose work if any company or other organization access user KYC then Blockchain will send EMAIL notification to KYC user about access.

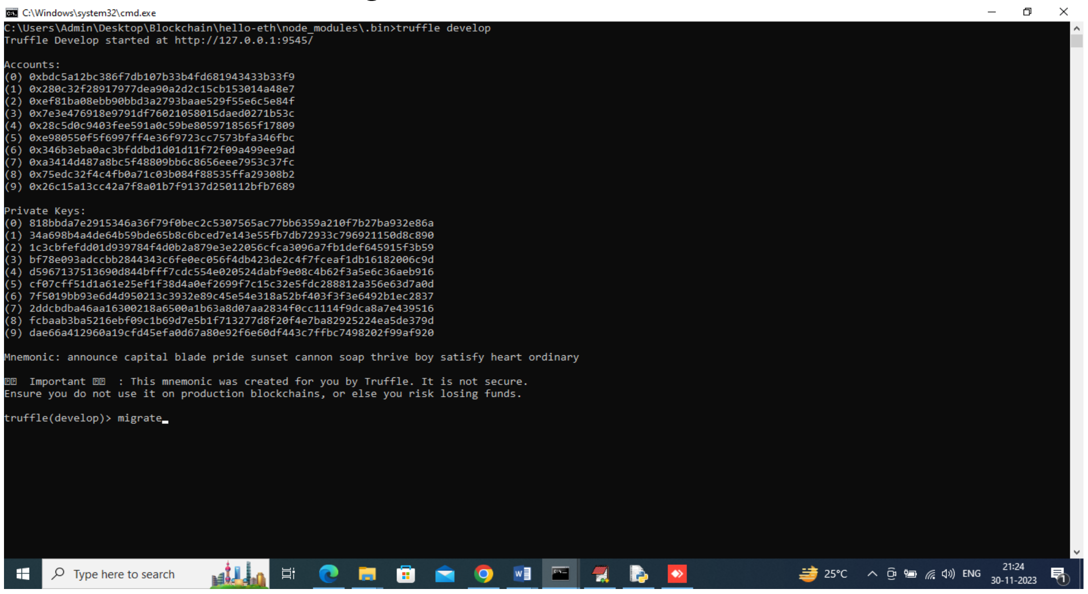
Above advantages of Blockchain diverting us to manage all KYC records using Blockchain. Blockchain can store or retrieve data using Smart Contract which can be designed using Solidity code. Smart Contract contains functions which will be access by Blockchain to manage user data. In below screen we are showing smart contract code



```
pragma solidity >= 0.8.11 <= 0.8.11;

contract KYCContract {
   string public user_registration;
   string public kyc;


   //call this function to register user details data to Blockchain
   function setSignup(string memory ur) public {
      user_registration = ur;
   }
  //get register details
   function getSignup() public view returns (string memory) {
      return user_registration;
   }

   //call this function to save KYC details in Blockchain
   function setKYC(string memory ky) public {
      kyc = ky;
   }
  //get KYC details
   function getKYC() public view returns (string memory) {
      return kyc;
   }

  constructor() public {
      user_registration="";
```

In above contract we have defined function to manage user sign up details and KYC details and this contract need to deploy in Blockchain using below steps

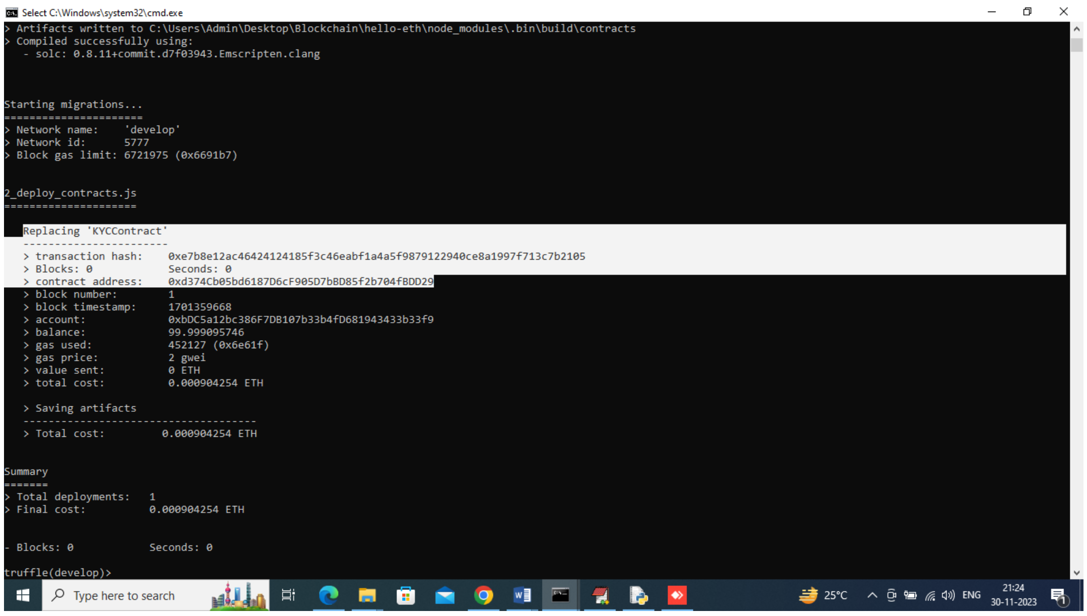
1) First go inside ‘hello-eth/node-modules/bin’ folder and then find for ‘runBlockchain.bat’ file and then double click on that file to get below Ethereum screen



```
C:\Users\Admin\Desktop\Blockchain\hello-eth\node_modules\.bin>truffle develop
Truffle Develop started at http://127.0.0.1:9545/

Accounts:
(0) 0xbdc5a12bc386f7db107b33b4fd681943433b33f9
(1) 0x280c32f28917977dea90a2d2c15cb153014a48e7
(2) 0xef81ba08ebb90bbd3a2793baae529f55e6c5e84f
(3) 0x7e3e476918e9791df76021058015daed0271b53c
(4) 0x28c5d0c9403fee591a0c59be8059718565f17809
(5) 0xe980550f5f6997ff4e36f9723cc7573bfa346fbc
(6) 0x346b3eba0ac3bfddbd1d01d11f72f09a499ee9ad
(7) 0xa3414d487a8bc5f48809bb6c8656eee7953c37fc
(8) 0x75edc32f4c4fb0a71c03b084f88535ffa29308b2
(9) 0x26c15a13cc42a7f8a01b7f9137d250112bfb7689

Private Keys:
(0) 818bbda7e2915346a36f79f0bec2c5307565ac77bb6359a210f7b27ba932e86a
(1) 34a698b4a4de64b59bde65b8c6bced7e143e55fb7db72933c796921150d8c890
(2) 1c3cbfefdd01d939784f4d0b2a879e3e22056cfca3096a7fb1def645915f3b59
(3) bf78e093adccbb2844343c6fe0ec056f4db423de2c4f7fceaf1db16182006c9d
(4) d5967137513690d844bfff7cdc554e020524dabf9e08c4b62f3a5e6c36aeb916
(5) cf07cff51d1a61e25ef1f38d4a0ef2699f7c15c32e5fdc288812a356e63d7a0d
(6) 7f5019bb93e6d4d950213c3932e89c45e54e318a52bf403f3f3e6492b1ec2837
(7) 2ddcbdba46aa16300218a6500a1b63a8d07aa2834f0cc1114f9dca8a7e439516
(8) fcbaab3ba5216ebf09c1b69d7e5b1f713277d8f20f4e7ba82925224ea5de379d
(9) dae66a412960a19cfd45efa0d67a80e92f6e60df443c7ffbc7498202f99af920

Mnemonic: announce capital blade pride sunset cannon soap thrive boy satisfy heart ordinary

⚠️  Important ⚠️  : This mnemonic was created for you by Truffle. It is not secure.
Ensure you do not use it on production blockchains, or else you risk losing funds.

truffle(develop)> migrate
```

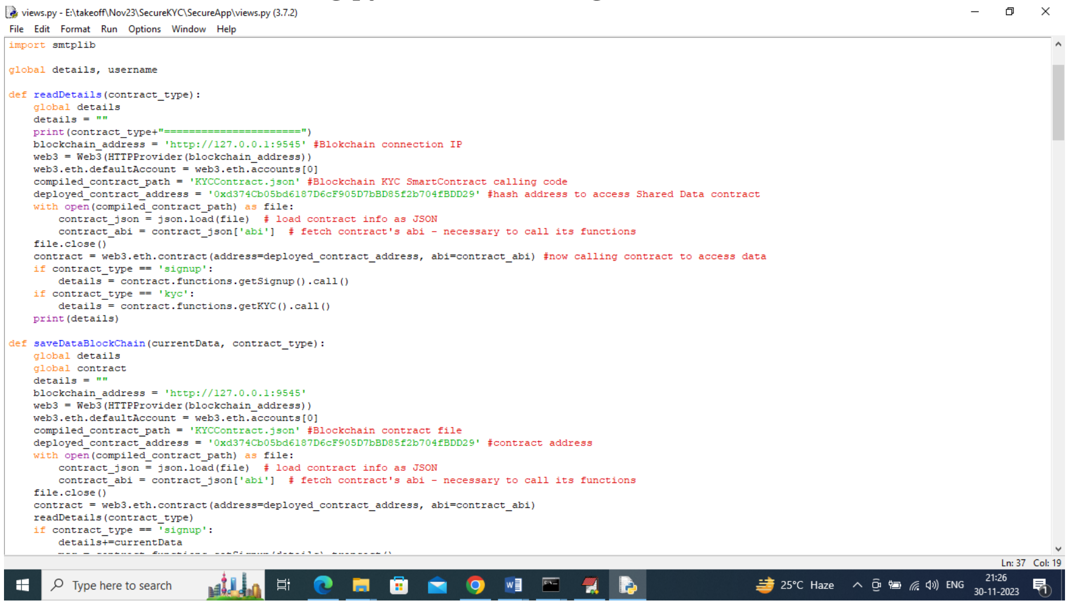
2)

3) In above screen Ethereum started with default accounts and private keys and then type command as ‘migrate’ and then press enter key to deploy contract and get below screen



```
Select C:\Windows\system32\cmd.exe
> Artifacts written to C:\Users\Admin\Desktop\Blockchain\hello-eth\node_modules\.bin\build\contracts
> Compiled successfully using:
   - solc: 0.8.11+commit.d7f03943.Emscripten.clang


Starting migrations...
======================
> Network name:    'develop'
> Network id:      5777
> Block gas limit: 6721975 (0x6691b7)


2_deploy_contracts.js
=====================

   Replacing 'KYCContract'
   -----------------------
   > transaction hash:    0xe7b8e12ac46424124185f3c46eabf1a4a5f9879122940ce8a1997f713c7b2105
   > Blocks: 0            Seconds: 0
   > contract address:    0xd374Cb05bd6187D6cF905D7bBD85f2b704fBDD29
   > block number:        1
   > block timestamp:     1701359668
   > account:             0xbDC5a12bc386F7DB107b33b4fD681943433b33f9
   > balance:             99.999095746
   > gas used:            452127 (0x6e61f)
   > gas price:           2 gwei
   > value sent:          0 ETH
   > total cost:          0.000904254 ETH

   > Saving artifacts
   -------------------------------------
   > Total cost:         0.000904254 ETH


Summary
=======
> Total deployments:   1
> Final cost:          0.000904254 ETH


- Blocks: 0            Seconds: 0

truffle(develop)>
```

4)
5) In above screen in white colour text can see KY Contract deployed and got contract address also and this address need to specify in PYTHON code to access that contract to store and get user data. In below screen showing python code calling smart contract



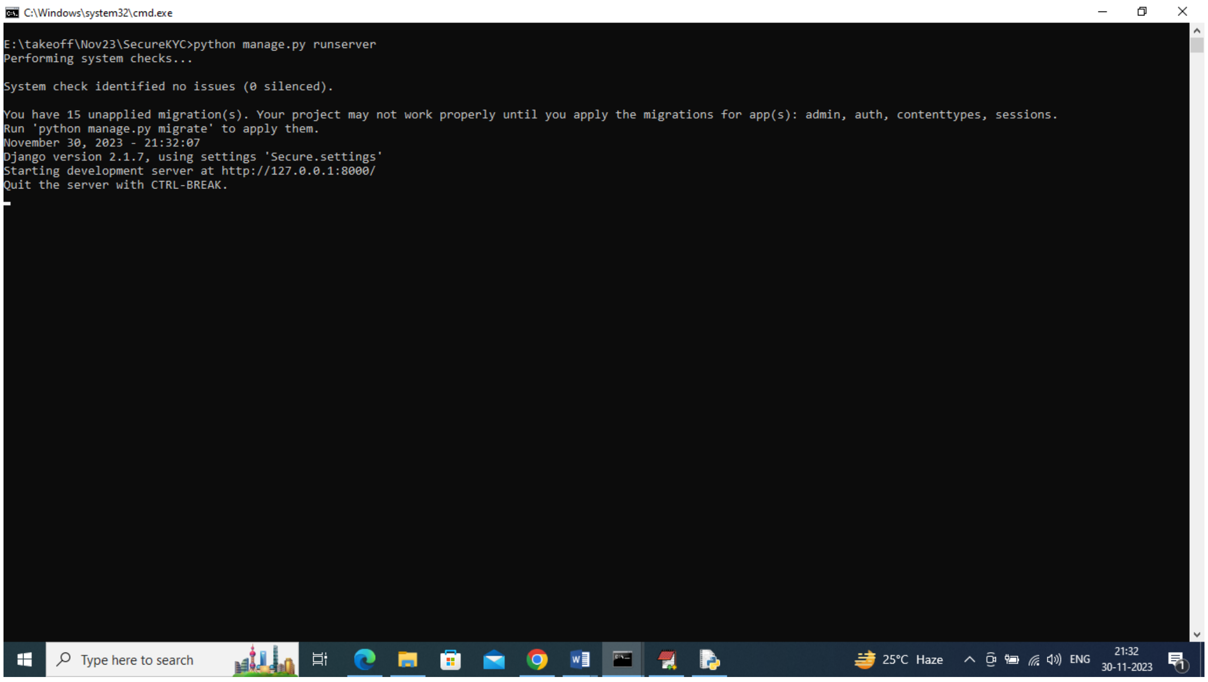
```
views.py - E:\takeoff\Nov23\SecureKYC\SecureApp\views.py (3.7.2)
File  Edit  Format  Run  Options  Window  Help
import smtplib

global details, username

def readDetails(contract_type):
    global details
    details = ""
    print(contract_type+"======================")
    blockchain_address = 'http://127.0.0.1:9545' #Blokchain connection IP
    web3 = Web3(HTTPProvider(blockchain_address))
    web3.eth.defaultAccount = web3.eth.accounts[0]
    compiled_contract_path = 'KYCContract.json' #Blockchain KYC SmartContract calling code
    deployed_contract_address = '0xd374Cb05bd6187D6cF905D7bBD85f2b704fBDD29' #hash address to access Shared Data contract
    with open(compiled_contract_path) as file:
        contract_json = json.load(file)  # load contract info as JSON
        contract_abi = contract_json['abi']  # fetch contract's abi - necessary to call its functions
    file.close()
    contract = web3.eth.contract(address=deployed_contract_address, abi=contract_abi) #now calling contract to access data
    if contract_type == 'signup':
        details = contract.functions.getSignup().call()
    if contract_type == 'kyc':
        details = contract.functions.getKYC().call()
    print(details)

def saveDataBlockChain(currentData, contract_type):
    global details
    global contract
    details = ""
    blockchain_address = 'http://127.0.0.1:9545'
    web3 = Web3(HTTPProvider(blockchain_address))
    web3.eth.defaultAccount = web3.eth.accounts[0]
    compiled_contract_path = 'KYCContract.json' #Blockchain contract file
    deployed_contract_address = '0xd374Cb05bd6187D6cF905D7bBD85f2b704fBDD29' #contract address
    with open(compiled_contract_path) as file:
        contract_json = json.load(file)  # load contract info as JSON
        contract_abi = contract_json['abi']  # fetch contract's abi - necessary to call its functions
    file.close()
    contract = web3.eth.contract(address=deployed_contract_address, abi=contract_abi)
    readDetails(contract_type)
    if contract_type == 'signup':
        details+=currentData
```

6)
7) In above screen read red colour comments to know about contract calling and now contract deployed and let that contract running.

To implement this project we have designed following modules

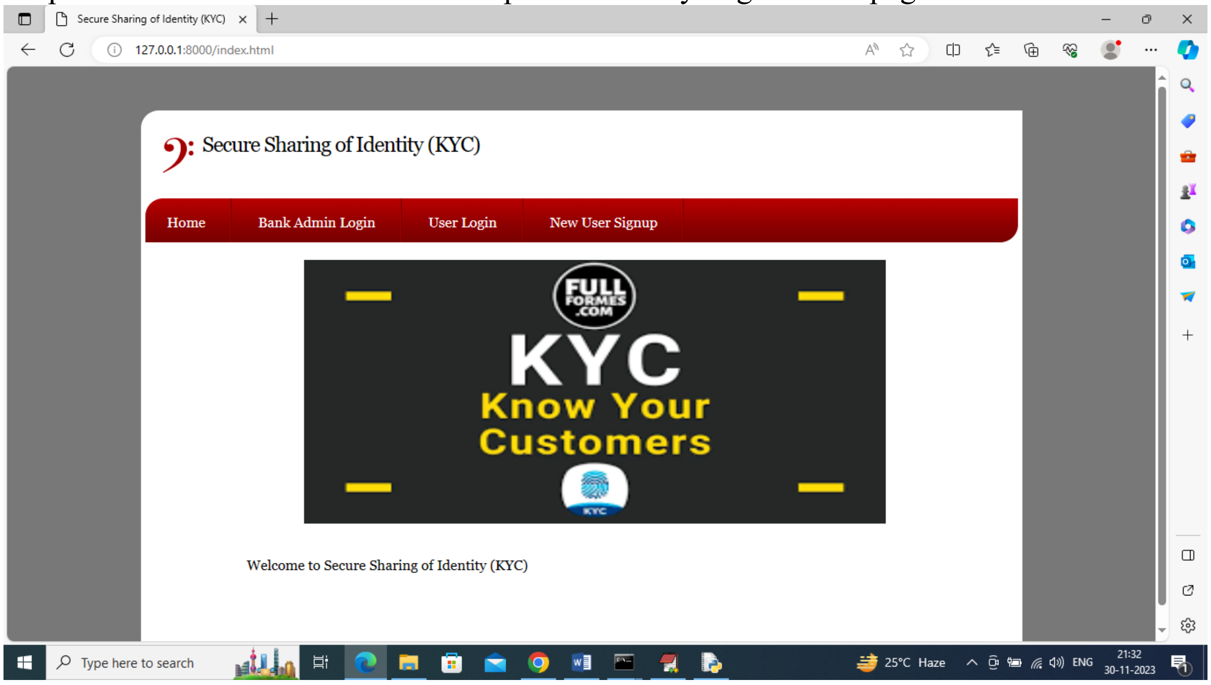
1) User module: user can sign up with the application and then login and can upload KYC details with Identity proof image and can check status of KYC and can get notification on Registered EMAIL about KYC access by which organization
2) Band Admin Module: this user can sign up and login to application and then can view list of uploaded KYC and after all verification Bank Admin may Accept or Reject user KYC. All this KYC can be access by only those Bank Admin who registered with Blockchain and if any organization access any User KYC then Blockchain will forward mail notification.

To run project double click on 'runServer.bat' file to start python server and get below page



```
E:\takeoff\Nov23\SecureKYC>python manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).

You have 15 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
November 30, 2023 - 21:32:07
Django version 2.1.7, using settings 'Secure.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

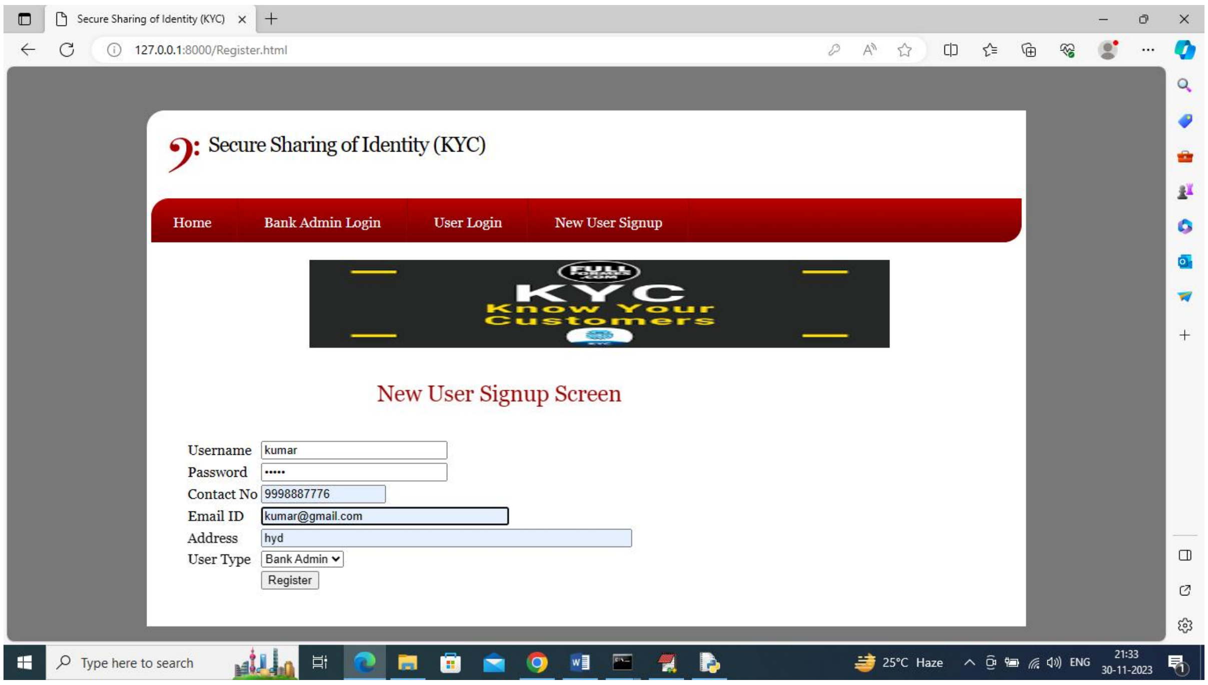
In above screen python server started and now open browser and enter URL as ‘http://127.0.0.1:8000/index.html’ and press enter key to get below page



Secure Sharing of Identity (KYC)

Welcome to Secure Sharing of Identity (KYC)

In above screen click on ‘New User Signup’ to add users to Blockchain and get below page



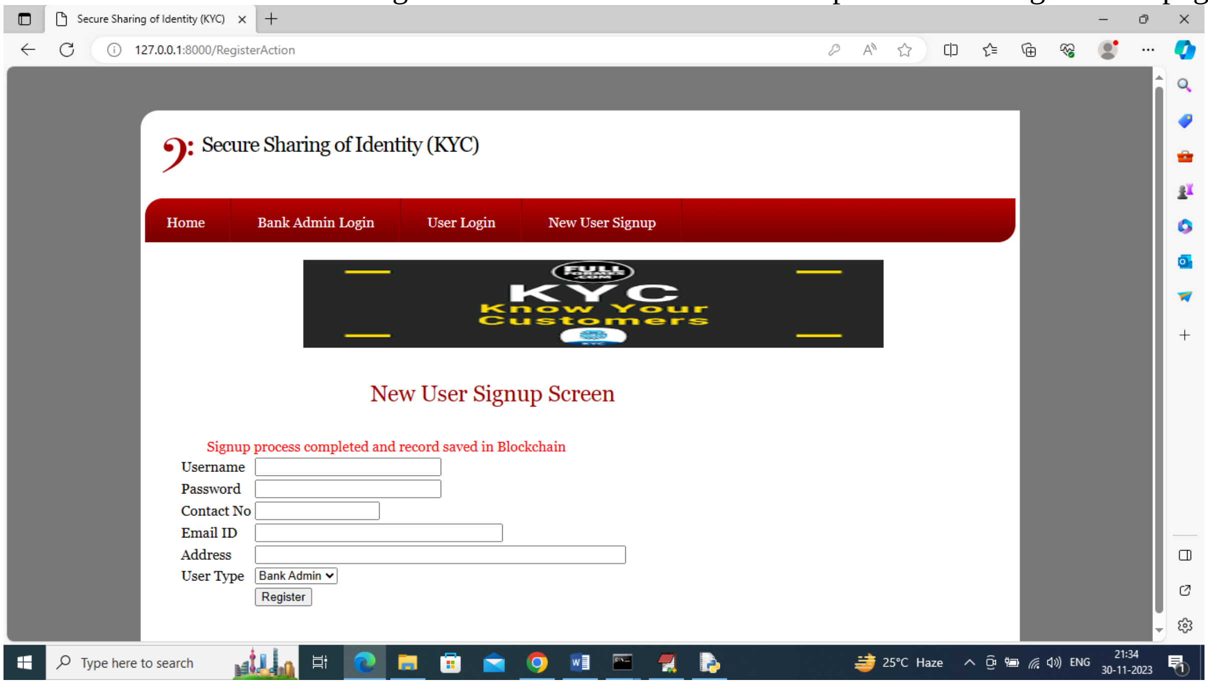
Secure Sharing of Identity (KYC)

New User Signup Screen

Username	kumar
Password	•••••
Contact No	9998887776
Email ID	kumar@gmail.com
Address	hyd
User Type	Bank Admin

Register

In above screen user is entering details as Bank Admin and then press button to get below page



Secure Sharing of Identity (KYC)

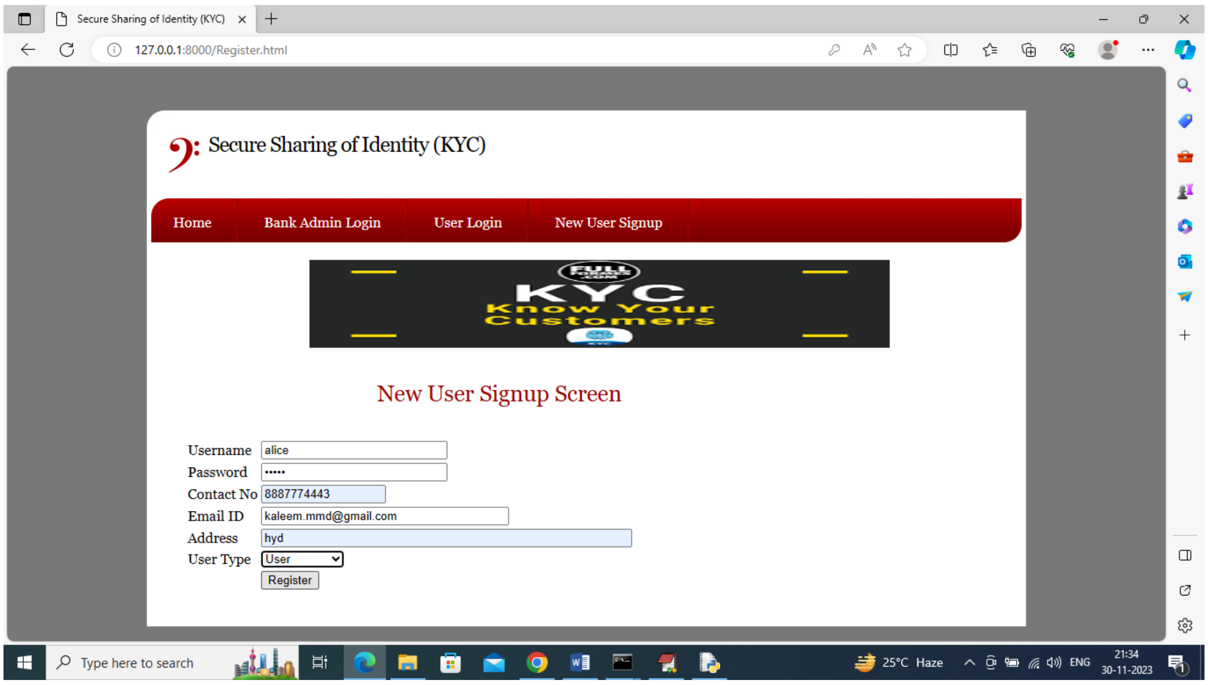
New User Signup Screen

Signup process completed and record saved in Blockchain

Username	
Password	
Contact No	
Email ID	
Address	
User Type	Bank Admin

Register

In above screen Bank Admin sign up completed and similarly add one user like below screen

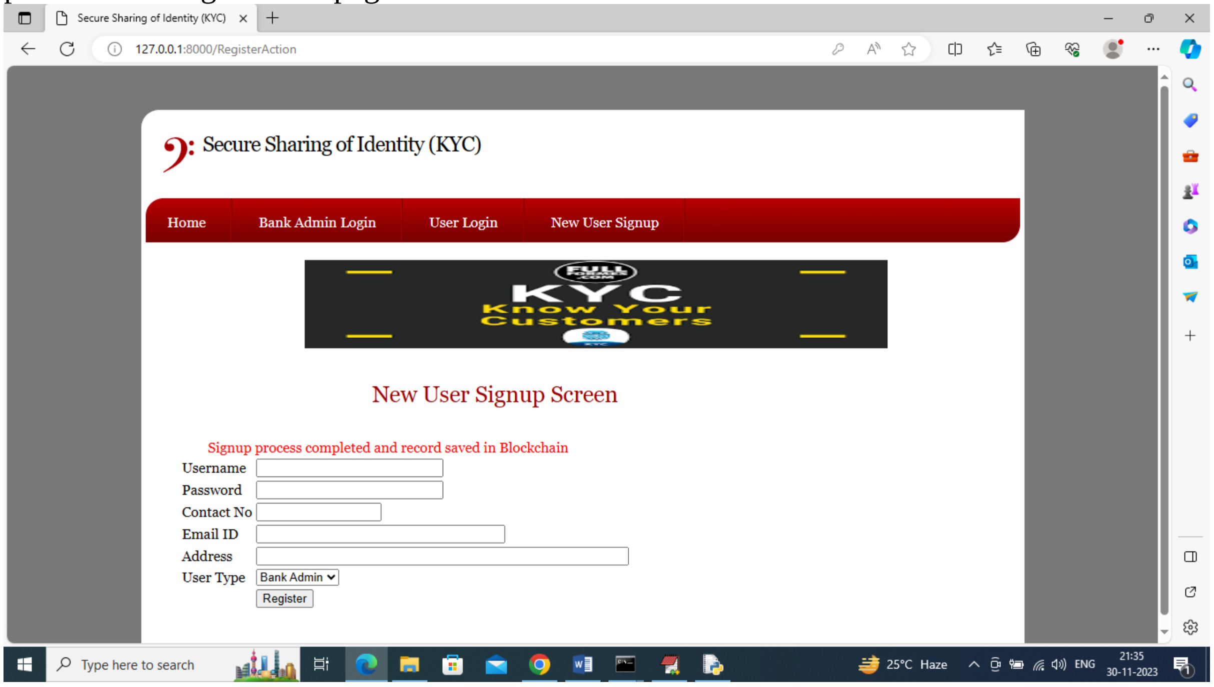


New User Signup Screen

Username	alice
Password	•••••
Contact No	8887774443
Email ID	kaleem.mmd@gmail.com
Address	hyd
User Type	User

Register

In above screen adding another details as ‘User’ and enter valid email ID to get notification and then press button to get below page



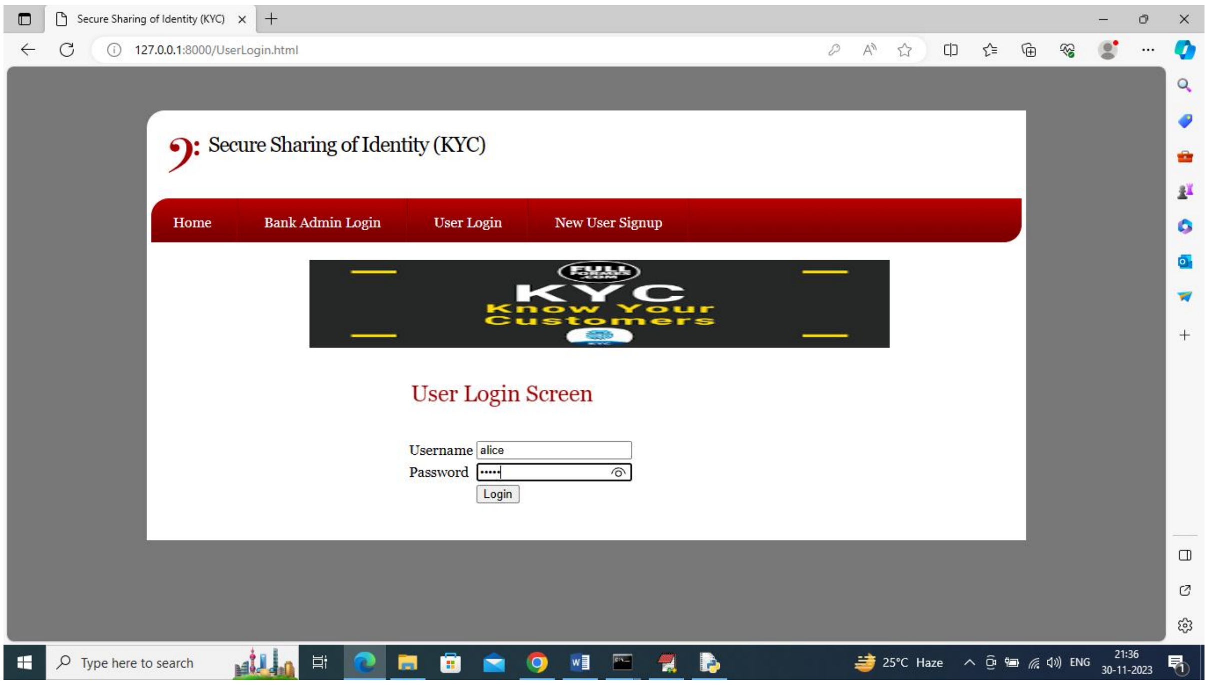
New User Signup Screen

Signup process completed and record saved in Blockchain

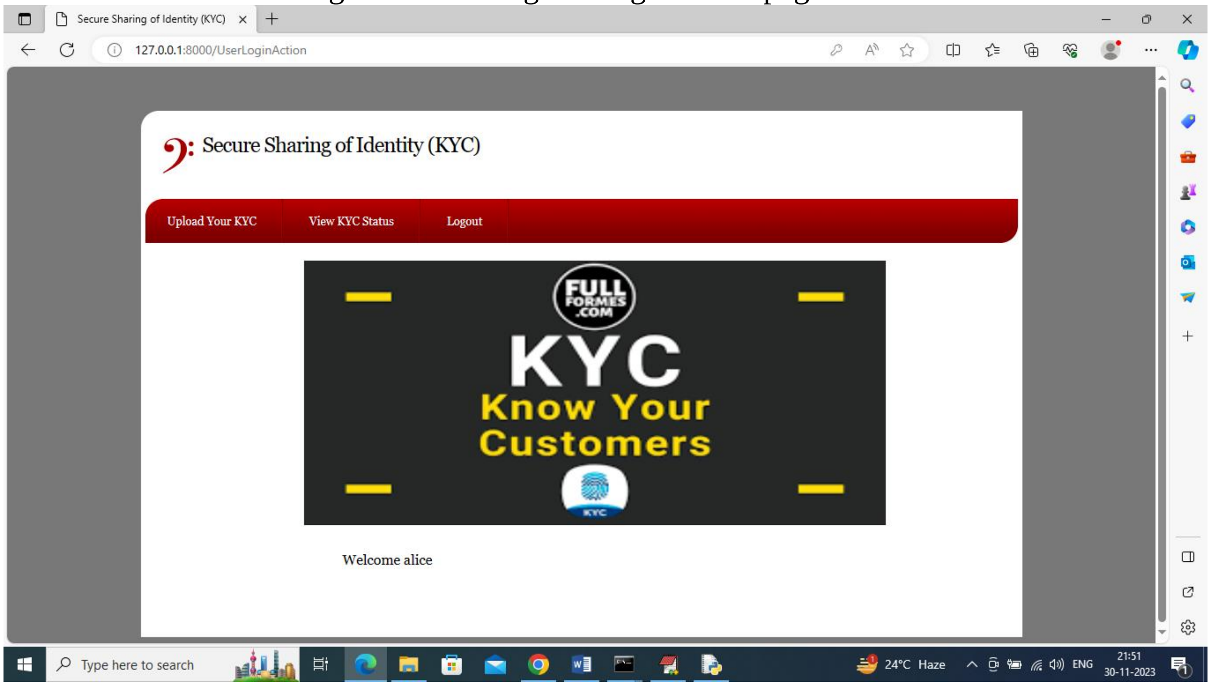
Username	
Password	
Contact No	
Email ID	
Address	
User Type	Bank Admin

Register

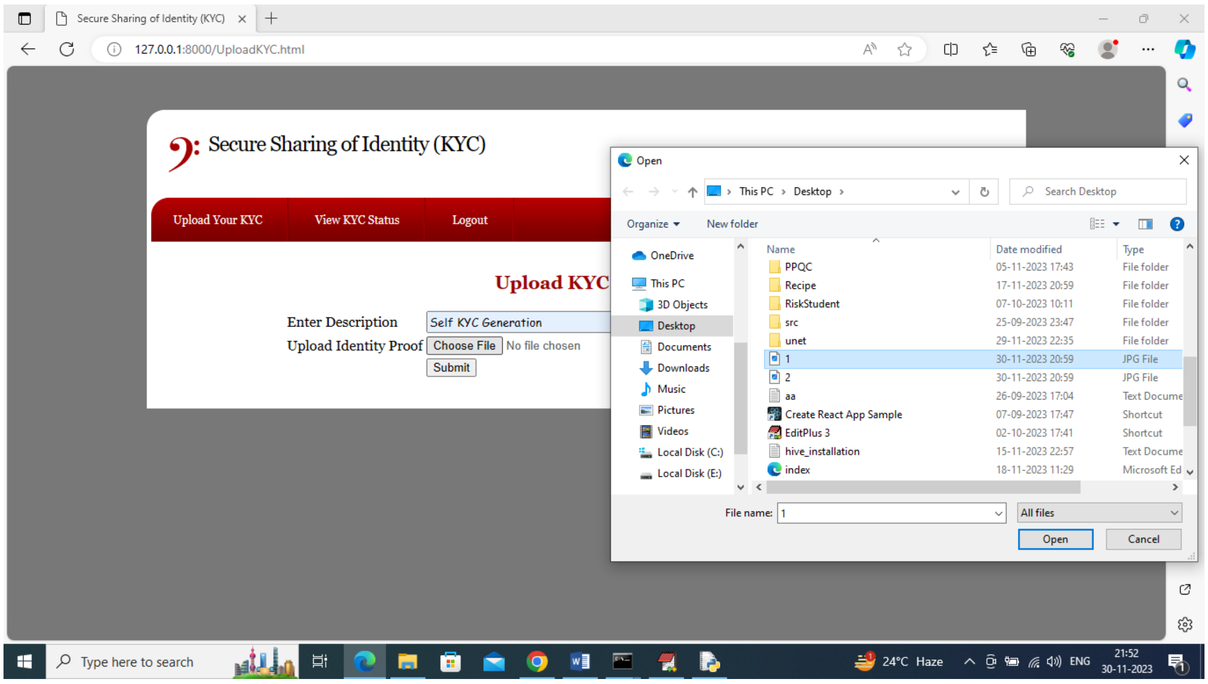
In above screen user sign up also completed and now click on ‘User Login’ link to get below login screen



In above screen user is login and after login will get below page



In above screen user can click on 'Upload Your KYC' link to upload KYC and get below page



Upload KYC

Enter Description: Self KYC Generation

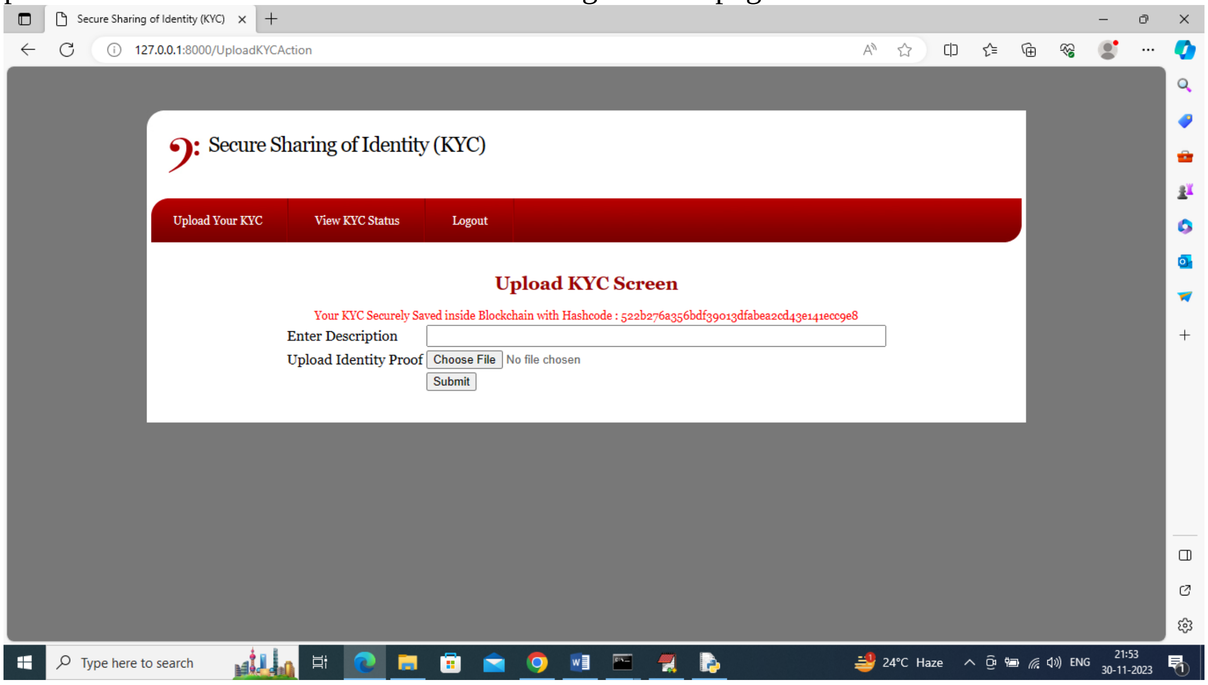
Upload Identity Proof: Choose File No file chosen

Submit

Name	Date modified	Type
PPQC	05-11-2023 17:43	File folder
Recipe	17-11-2023 20:59	File folder
RiskStudent	07-10-2023 10:11	File folder
src	25-09-2023 23:47	File folder
unet	29-11-2023 22:35	File folder
1	30-11-2023 20:59	JPG File
2	30-11-2023 20:59	JPG File
aa	26-09-2023 17:04	Text Docume
Create React App Sample	07-09-2023 17:47	Shortcut
EditPlus 3	02-10-2023 17:41	Shortcut
hive_installation	15-11-2023 22:57	Text Docume
index	18-11-2023 11:29	Microsoft Ed

File name: 1

In above screen user can enter KYC details and then upload some type of identity proof and then press button to save KYC to Blockchain and get below page


Secure Sharing of Identity (KYC)
127.0.0.1:8000/UploadKYCAction
Upload Your KYC
View KYC Status
Logout
Type here to search

Upload KYC Screen

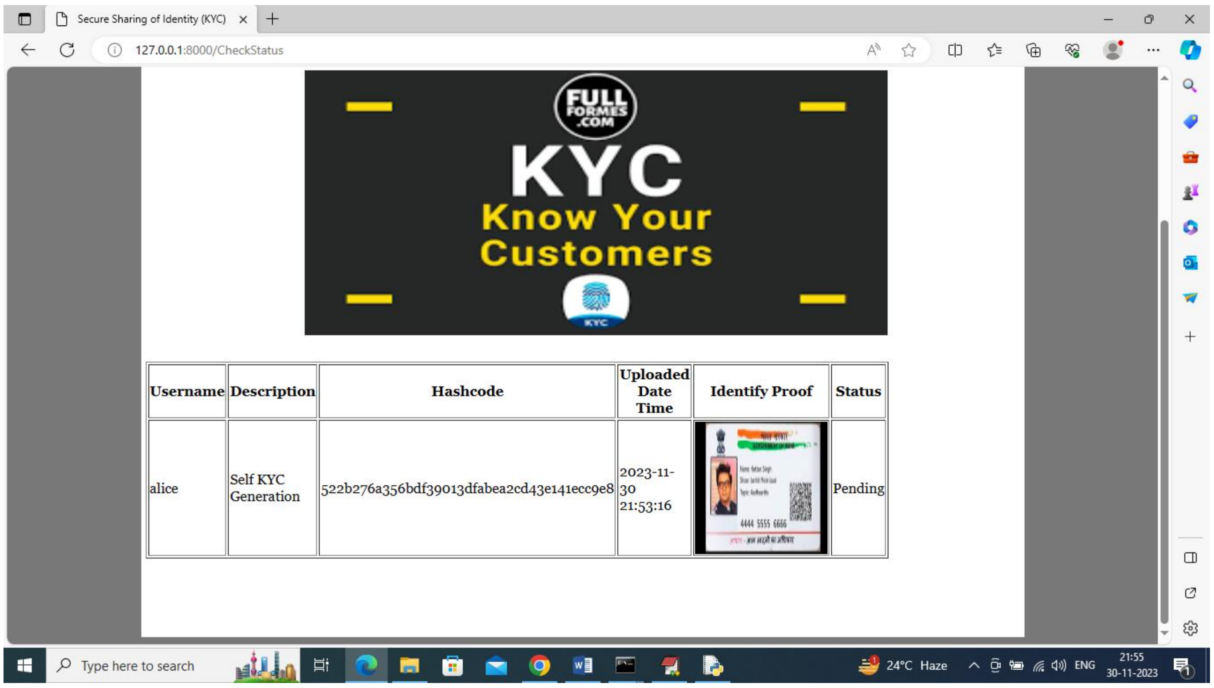
Your KYC Securely Saved inside Blockchain with Hashcode : 522b276a356bdf39013dfabea2cd43e141ecc9e8

Enter Description:

Upload Identity Proof: Choose File No file chosen

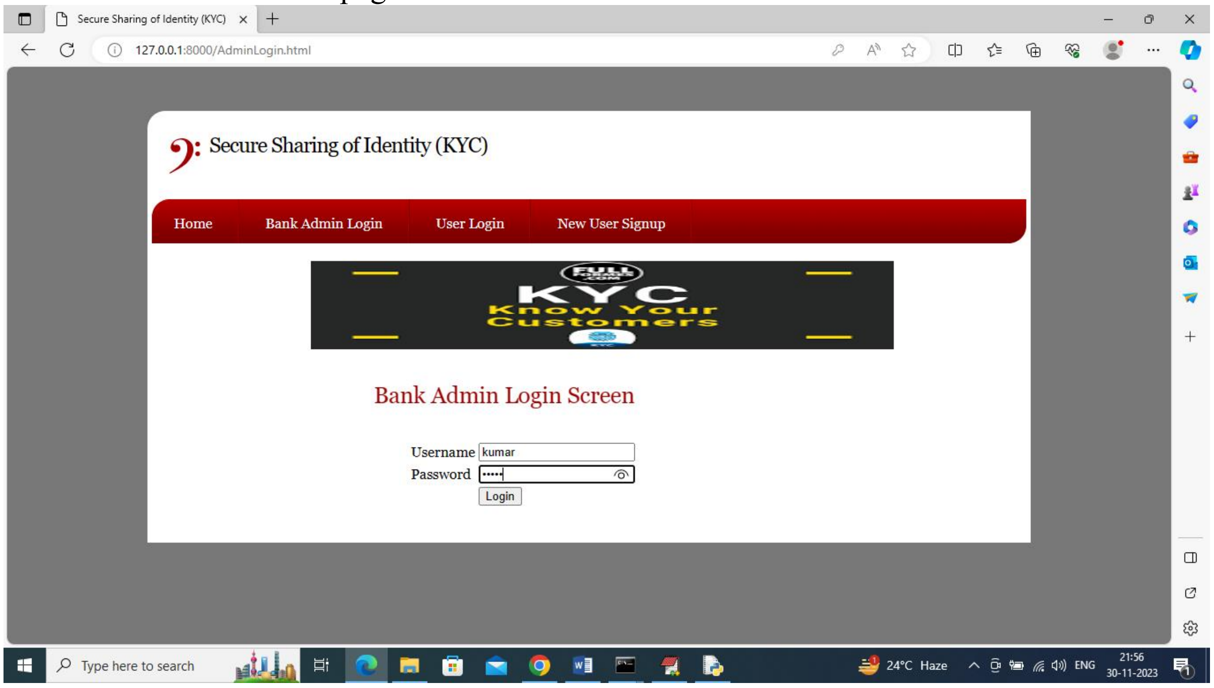
Submit

In above screen in red colour text can see KYC saved in Blockchain and displaying hash Blockchain address of KYC stored. Now KYC uploaded but not yet accepted by Bank Admin and now by click on ‘View KYC Status’ link to get below page



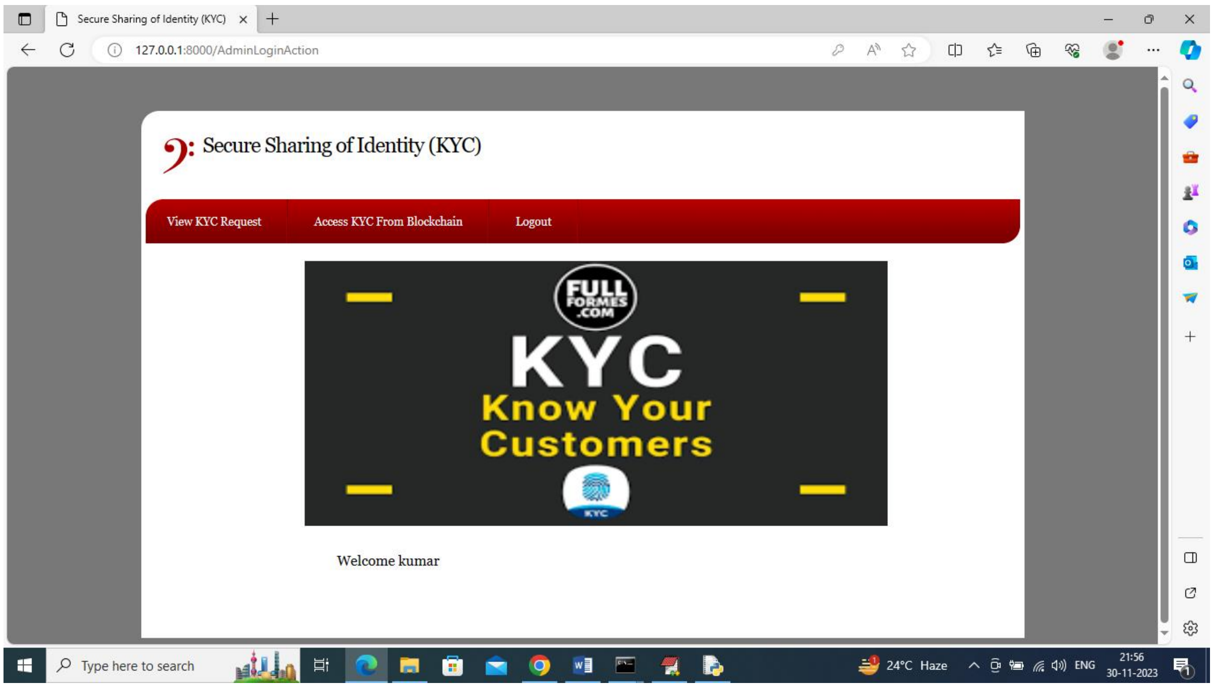
Username	Description	Hashcode	Uploaded Date Time	Identify Proof	Status
alice	Self KYC Generation	522b276a356bdf39013dfabea2cd43e141ecc9e8	2023-11-30 21:53:16		Pending

In above screen in last column user can see his KYC status as pending and now logout and login as 'Bank Admin' like below page



Username: kumar

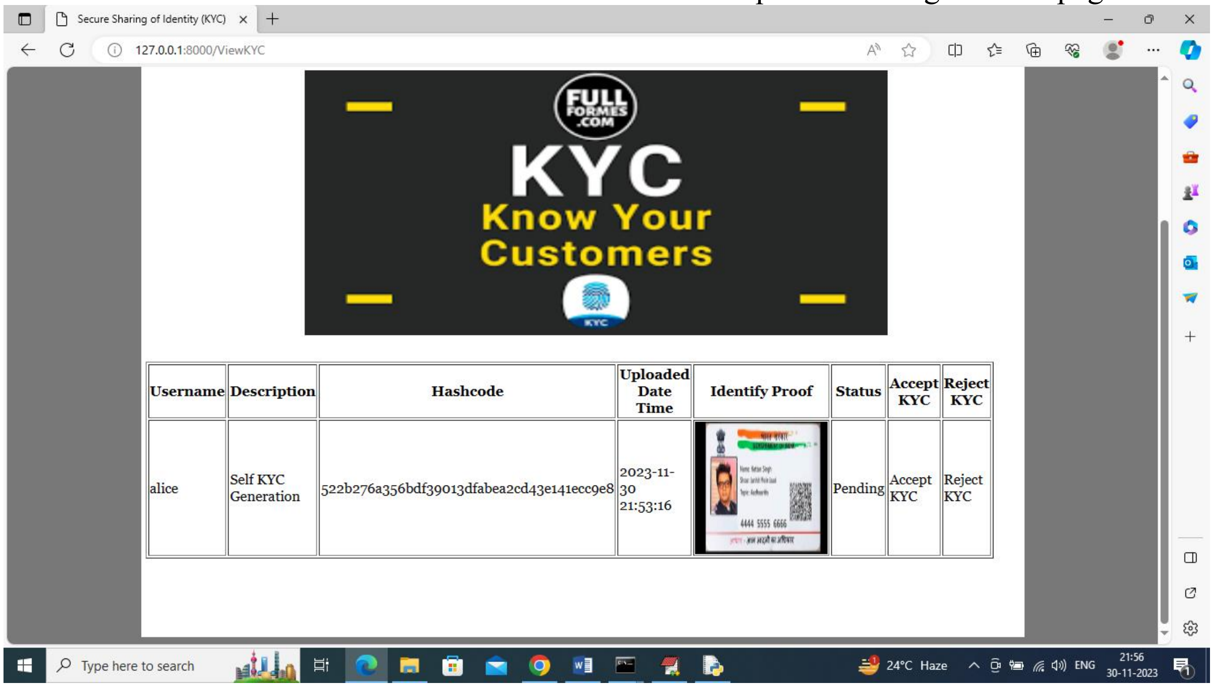
In above screen Bank Admin is login and after login will get below page


Secure Sharing of Identity (KYC)

127.0.0.1:8000/AdminLoginAction

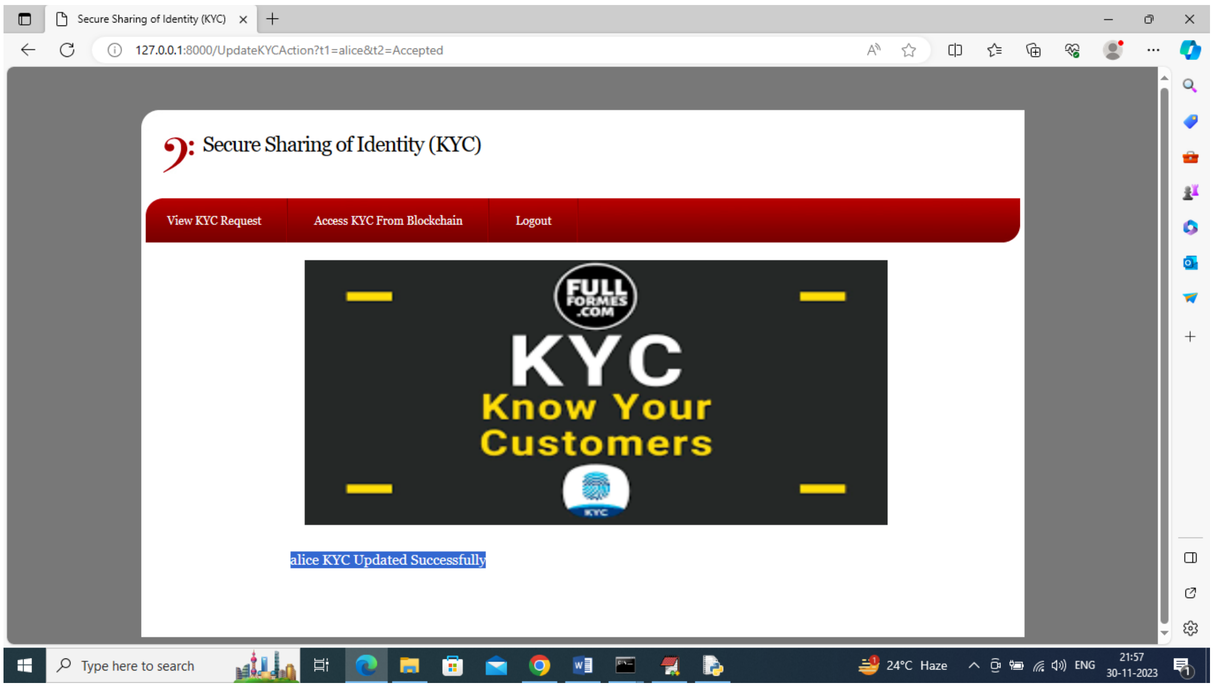
Welcome kumar

In above screen Bank Admin can click on ‘View KYC Request’ link to get below page


127.0.0.1:8000/ViewKYC

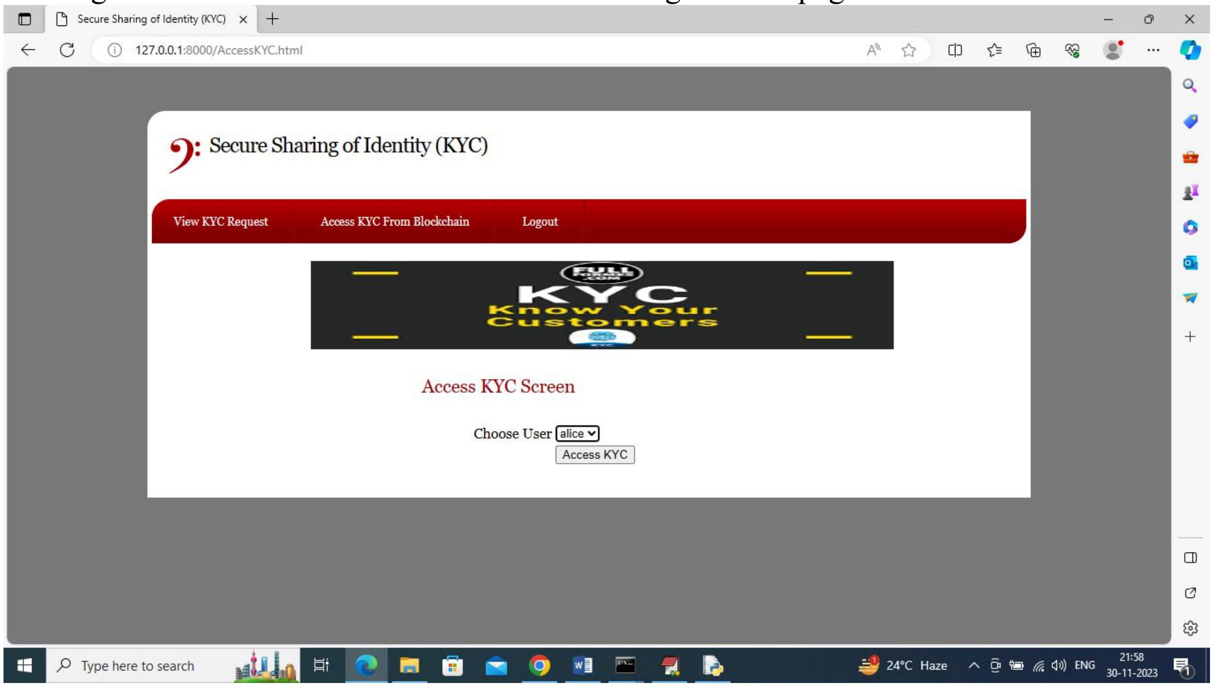
Username	Description	Hashcode	Uploaded Date Time	Identify Proof	Status	Accept KYC	Reject KYC
alice	Self KYC Generation	522b276a356bdf39013dfabea2cd43e141ecc9e8	2023-11-30 21:53:16		Pending	Accept KYC	Reject KYC

In above screen Bank Admin can see all details of user KYC and then click on either ‘Accept or Reject’ link to accept or reject KYC and get below page


Secure Sharing of Identity (KYC)
127.0.0.1:8000/UpdateKYCAction?t1=alice&t2=Accepted
Secure Sharing of Identity (KYC)
View KYC Request
Access KYC From Blockchain
Logout
FULL FORMES .COM
KYC
Know Your Customers
KYC
Type here to search
24°C Haze
ENG
21:57
30-11-2023

alice KYC Updated Successfully

In above screen can see KYC accepted successfully and any Bank Admin can access that KYC by clicking on 'Access KYC from Blockchain' link to get below page

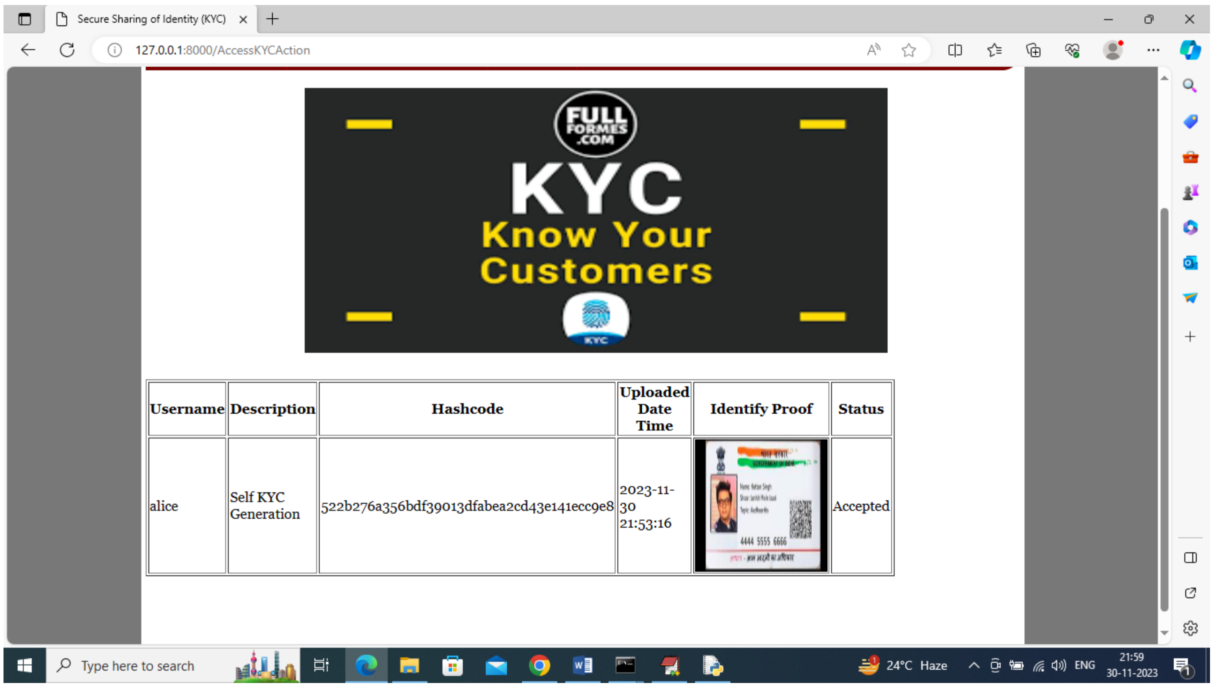

Secure Sharing of Identity (KYC)
127.0.0.1:8000/AccessKYC.html
Secure Sharing of Identity (KYC)
View KYC Request
Access KYC From Blockchain
Logout
KYC
Know Your Customers
Type here to search
24°C Haze
ENG
21:58
30-11-2023

Access KYC Screen

Choose User alice

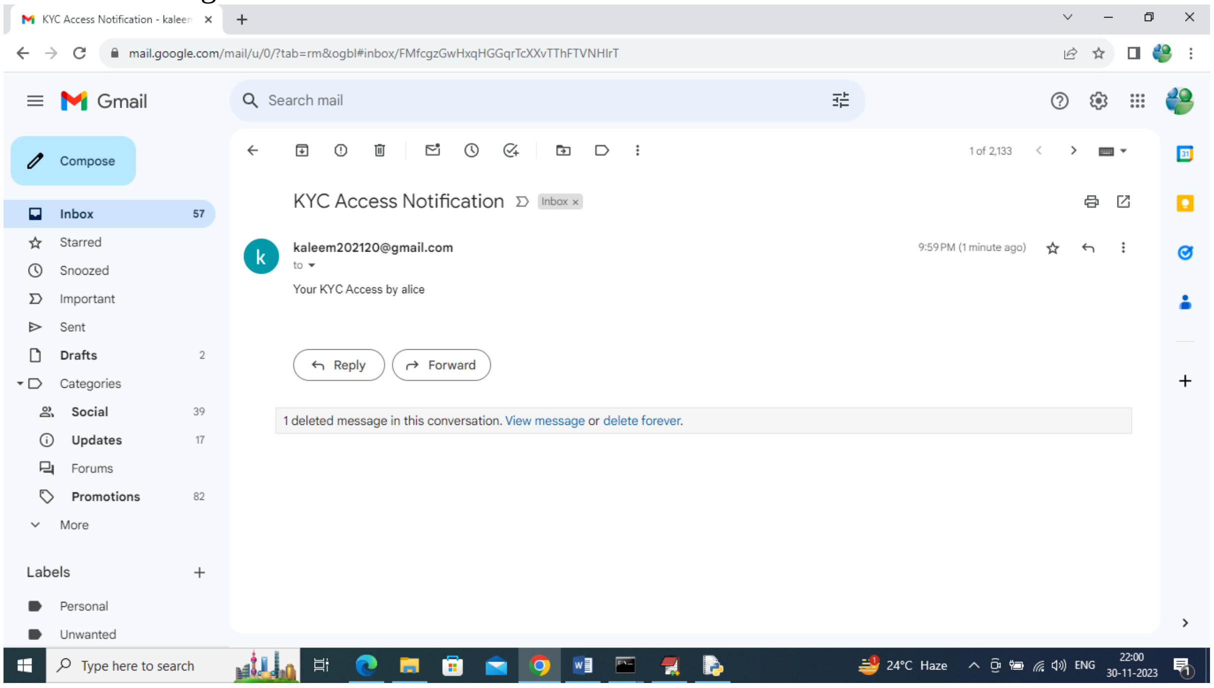
Access KYC

In above screen Bank Admin will select desired user and press button to access that KYC and get below output



Username	Description	Hashcode	Uploaded Date Time	Identify Proof	Status
alice	Self KYC Generation	522b276a356bdf39013dfabea2cd43e141ecc9e8	2023-11-30 21:53:16		Accepted

In above screen Bank Admin Kumar can view above KYC with all details and once he access then Alice user will get notification in mail like below screen



KYC Access Notification

kaleem202120@gmail.com
to

Your KYC Access by alice

1 deleted message in this conversation. View message or delete forever.

In above mail screen can see Alice user got notification.

So by following above screens we can saved all users KYC in Blockchain securely and only those users can access this KYC who has access control to Blockchain and if any Bank Admin access user KYC then user will get EMAIl notification

6. CONCLUSION AND FUTURE WORK

CONCLUSION

The "Secure Sharing of Identity (KYC)" project marks a paradigm shift in how identity information is managed and shared. By addressing the vulnerabilities in the existing system through the integration of blockchain technology and advanced notification mechanisms, the project strives to establish a secure, transparent, and user-centric approach to identity management. This innovation not only safeguards users against potential threats but also fosters trust in online identity verification processes. As we navigate the complexities of the digital age, the "Secure Sharing of Identity (KYC)" project stands as a pioneering effort to fortify the foundations of secure identity management in our interconnected world.

7. REFRENCES

1. **1** J. P. Moyano and O. Ross, "KYC optimization using distributed ledger technology", *Business & Information Systems Engineering*, vol. 59, no. 6, pp. 411-423, 2017.

2. N. K. Ostern and J. Riedel, "Know-Your-Customer (KYC) Requirements for Initial Coin Offerings", *Business & Information Systems Engineering*, pp. 1-17, 2020.

3. D. De Smet and A. L. Mention, "Improving auditor effectiveness in assessing KYC/AML practices: Case study in a Luxembourgish context", *Managerial Auditing Journal*, 2011.

4. R. Syah, M. K. Nasution, M. Elveny and H. Arbie, "Optimization model for customer behavior with MARS and KYC system", *Journal of Theoretical and Applied Information Technology*, vol. 98, no. 13, 2020.

5. I. Bashir, Mastering Blockchain: Distributed ledger technology decentralization and smart contracts explained, Packt Publishing Ltd, 2018.

6. G. Wood, "Ethereum: A secure decentralised generalised transaction ledger", *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1-32, 2014.

7. E. Androulaki et al., "Hyperledger fabric: a distributed operating system for permissioned blockchains", *Proceedings of the thirteenth EuroSys conference*, pp. 1-15, 2018.

8. W. Shbair, M. Steichen and J. François, "Blockchain orchestration and experimentation framework: A case study of KYC", *IEEE/IFIP Man2Block 2018-IEEE/IFIP Network Operations and Management Symposium*, 2018.

9. N. Sundareswaran, S. Sasirekha, I. J. L. Paul, S. Balakrishnan and G. Swaminathan, "Optimised KYC Blockchain System", *2020 International Conference on Innovative Trends in Information Technology (ICITIIT),* pp. 1-6, 2020.

10.S. Nakamoto, "Re: Bitcoin P2P e-cash paper", *The Cryptography Mailing List*, 2008.