

Research Article

Secure Sharing of Identity (KYC)

Pathepuram Mamatha¹, Majhi Sneha², Thumkurta Ashray Reddy³, Danthuri Chandrakanth⁴ and V. Shiva Narayana Reddy^{5,*}

^{1,2,3,4}Student, Department of CSE, Samskurthi College of Engineering and Technology, 500001, Hyderabad, Telangana, India

⁵Asst Professor, Department of CSE, Samskurthi College of Engineering and Technology, Hyderabad, 500001, Telangana, India

*Corresponding Author: V. Shiva Narayana Reddy. Email: jayanthi515msk@gmail.com

Received: 29/07/2025; Revised: 30/08/2025; Accepted: 25/09/2025; Published: 30/09/2025.

DOI: <https://doi.org/10.69996/fmep.2025021>

Abstract: In the digital era, identity verification processes such as Know Your Customer (KYC) are critical for ensuring the legitimacy of users in financial and governmental systems. However, conventional KYC mechanisms suffer from centralized storage models, making sensitive user data vulnerable to breaches, unauthorized access, and misuse. This paper presents a blockchain-based solution for secure KYC data sharing, aiming to enhance privacy, transparency, and data control for users. Leveraging the decentralized and immutable nature of blockchain technology, the proposed system enables secure storage, retrieval, and access control of user identity data through smart contracts. The model integrates a notification mechanism that alerts users via email whenever their data is accessed, thereby reinforcing transparency and accountability. Through a series of simulations and user interface walkthroughs, the effectiveness of the model is demonstrated in enabling only authorized institutions to access KYC data while safeguarding it against tampering or leakage.

Keywords:Blockchain, KYC (Know Your Customer), Identity management, Data privacy, Smart contracts, Decentralized storage.

1.Introduction

The increasing digitization of services across industries such as finance, healthcare, telecommunications, and government has resulted in a significant reliance on digital identity verification systems. One of the most crucial identity verification protocols employed globally is Know Your Customer (KYC), which mandates the validation of personal identification information to ensure the authenticity of users, thereby mitigating the risk of fraud, identity theft, and money laundering [1], [2]. Regulatory bodies worldwide, including the Financial Action Task Force (FATF) and various national authorities, have emphasized the enforcement of KYC procedures as a preventive measure against financial crimes [3]. Traditional KYC systems are typically centralized, wherein users submit their personal documents and sensitive data to service providers for verification. These service providers, in turn, store and manage the data for compliance and record-keeping purposes. While effective to an extent, such systems raise numerous concerns about privacy, security, and data redundancy [4]. Users often have to submit the same documents repeatedly across different platforms, leading to unnecessary duplication and potential overexposure of personal data. Moreover, central repositories of sensitive KYC data are attractive targets for cybercriminals, resulting in large-scale data breaches and privacy violations [5]. The centralized nature of existing KYC infrastructures has raised fundamental questions about data ownership and control. Typically, users relinquish control over their identity information once it is submitted to service providers, who store the data in internal databases



without user-managed access permissions [6]. This lack of user-centric control violates key principles of data minimization and informed consent, which are foundational to modern data protection regulations such as the General Data Protection Regulation (GDPR) [7]. As data privacy awareness increases globally, users and regulators alike are demanding more transparent, secure, and user-controlled approaches to identity sharing. Inter-organizational KYC data sharing presents additional technical and regulatory challenges. In order to reduce compliance costs and eliminate redundant verification processes, financial institutions and service providers seek mechanisms to reuse KYC data with user consent across entities. However, such sharing must be conducted in a manner that ensures the data's integrity, confidentiality, and legality across jurisdictions [8]. Conventional file transfer protocols or APIs lack the necessary transparency and immutability to guarantee trust among competing organizations or to provide auditability for compliance assessments. Moreover, the scalability and efficiency of existing KYC mechanisms are increasingly under pressure due to the growing number of digital users and the rising complexity of identity data. Manual verification processes are labor-intensive and costly, while even automated approaches often involve significant back-end infrastructure and human oversight [9]. With the proliferation of digital wallets, remote onboarding, and mobile-first services, the demand for real-time, secure, and privacy-preserving KYC verification continues to grow.

To address these limitations, various researchers and industry bodies have explored the use of digital identity frameworks, decentralized identity solutions, and advanced cryptographic tools to improve KYC data sharing mechanisms [10], [11]. These approaches aim to shift control back to users, enabling them to selectively share their identity attributes without revealing unnecessary personal details. Such solutions have the potential to not only enhance security and user trust but also streamline onboarding procedures and reduce operational costs for service providers. Another layer of complexity arises from regulatory compliance across multiple jurisdictions. KYC processes are governed by diverse and sometimes conflicting legal frameworks, especially in cross-border scenarios. For instance, what is deemed sufficient identity proof in one country may not meet the legal standards in another. This lack of standardization hampers interoperability and increases compliance burdens [12]. Furthermore, ensuring the legality of digital signatures, maintaining verifiable audit trails, and managing consent records in a tamper-proof manner are all pressing concerns that current solutions only partially address. The rise in identity fraud cases also highlights the insufficiency of existing KYC mechanisms in safeguarding against impersonation and forged documentation [13]. Fraudsters exploit weak document verification practices and system loopholes to bypass traditional defenses, resulting in significant financial and reputational losses for institutions. In response, researchers have emphasized the need for tamper-evident data storage, robust user authentication methods, and real-time monitoring to prevent misuse and alert stakeholders of suspicious activities [14]. The evolution of digital identity is also being influenced by the concept of Self-Sovereign Identity (SSI), which advocates for individual ownership and control of identity data. In an SSI model, users maintain cryptographic proofs of their identity on their devices and share them with third parties without relying on a central authority [15]. While promising, implementing SSI at scale presents technological, social, and legal challenges, particularly in integrating with legacy KYC infrastructure and achieving broad institutional acceptance.

This paper contributes a novel, secure, and transparent framework for managing and sharing Know Your Customer (KYC) data using blockchain technology. Unlike traditional centralized KYC systems that are prone to data breaches and unauthorized access, the proposed

model ensures decentralized, tamper-proof storage of sensitive identity information. By integrating smart contracts, the system automates access control and enforces predefined data-sharing policies, eliminating the need for intermediaries. A unique feature of the model is its built-in email notification mechanism, which alerts users whenever their KYC data is accessed by registered organizations, thereby enhancing user trust and accountability. Additionally, the paper provides a comprehensive implementation architecture, including Ethereum deployment, Solidity-based contract development, and a user interface with Python integration. The system is modular, allowing scalability across financial and governmental institutions, and sets a strong foundation for future advancements in secure digital identity verification[16-17].

2. Blockchain Hashcode Model for Data Security

In the contemporary digital landscape, approximately 70% of user data resides online, often stored on centralized servers that are vulnerable to manipulation, leakage, or unauthorized access by administrators or third-party actors. Users are frequently compelled to disclose confidential information such as their name, address, ZIP code, gender, and even medical history during identity verification or Know Your Customer (KYC) processes. However, these centralized infrastructures lack transparency, and users are seldom informed of when or how their personal information is accessed or used. To address these critical issues, we propose a Blockchain-based KYC Sharing system that ensures decentralized storage, cryptographic protection, and user-controlled data access mechanisms. The foundation of the proposed system lies in the immutability and transparency of the blockchain. Each KYC record is encapsulated in a transaction, which is then hashed and added to a chain of blocks using a cryptographic hash function. Let the original KYC data of a user be denoted as Du . A secure hash $H(Du)$ is generated using a cryptographic function, typically SHA-256 defined in equation (1)

$$H(Du) = SHA - 256(Du) \quad (1)$$

This hash is appended to the blockchain along with metadata such as a timestamp T , the unique user ID Ui , and access permissions P . The overall transaction stored in the blockchain is represented as in equation (2)

$$Tx_i = \{Ui, H(Du), T, P\} \quad (2)$$

When a new transaction is created or an organization requests access to a user's KYC data, the blockchain network validates the transaction by recalculating the hash of the previous block $Hi - 1$ and comparing it to the stored hash pointer. This ensures that any tampering with historical data would invalidate the entire chain. The block linkage is defined as in equation (3)

$$Bi = Hash(Bi - 1 \parallel Tx_i) \quad (3)$$

In equation (3) $\parallel$ denotes concatenation and Bi is the current block. This structure ensures the tamper-evident nature of blockchain records. Access to the KYC data is governed by smart contracts, which are coded in Solidity and deployed on the blockchain network. A smart contract ScS_cSc is triggered when a request is made by a verified organization Oj to access a user's identity information. The contract verifies the user's consent and the access permissions defined in PPP. Once the conditions are met, the contract allows read access and simultaneously triggers an EMAIL notification to the user, indicating that their KYC data was accessed by organization Oj at time Tj . This creates a transparent and auditable access trail. The smart contract also enforces role-based access control. Let Ar denote the access rights assigned to role r (e.g., "Bank", "Insurance", "Health Provider"). Access is granted only as stated in equation (4)

$$Oj \in RANDAr(Oj) \subseteq P(Ui) \quad (4)$$

In equation (4) R is the registered set of roles and (U_i) is the set of permissions granted by the user. This ensures fine-grained access control and compliance with privacy regulations. Additionally, to preserve privacy and prevent exposure of raw data on-chain, only encrypted KYC data or its hash is stored on the blockchain. The actual KYC documents are encrypted using the user's public key K_{pub} , and stored either off-chain (e.g., on IPFS or a secure cloud) with the encrypted address recorded on-chain defined in equation (5) and (6)

$$E(Du) = \text{Encrypt}_{K_{pub}}(Du) \quad (5)$$

$$\text{On-chain entry: } \{\text{IPFS_hash}(E(Du)), H(Du)\} \quad (6)$$

The proposed system leverages the cryptographic assurance, decentralized structure, and automated transparency of blockchain to address the core issues of centralized KYC storage. The use of hash functions ensures data immutability, while smart contracts enforce conditional access with real-time notifications, empowering users with visibility and control over their personal data. This privacy-preserving model transforms KYC into a secure, user-centric, and auditable framework suitable for modern digital ecosystems.

3.Hashcode for the Security

In the proposed Blockchain-based KYC system, Hashcodes play a critical role in ensuring the integrity, authenticity, and immutability of sensitive user data. A Hashcode is a unique, fixed-length string generated by applying a cryptographic hash function (such as SHA-256) to a piece of input data. When a user's KYC document or identity record is created or updated, it is first converted into a digital format and passed through the hash function to generate a unique hash value. This Hashcode, denoted $H(Du)$, where Du is the user's data, acts like a digital fingerprint. Even a single character change in the original data will produce a completely different hash, making unauthorized modifications instantly detectable. The blockchain stores only the hash, not the raw data, thus preserving privacy while ensuring data integrity. When a new block is added, it includes the Hashcode of the previous block, forming a linked chain of trust. If any block is tampered with, the hash linkage breaks, and the system immediately flags the inconsistency. This chaining mechanism ensures that once data is recorded, it cannot be altered without detection, providing tamper-evidence and non-repudiation. Additionally, during verification, the system recalculates the hash of the stored data and compares it with the existing Hashcode. A match confirms data integrity, while a mismatch indicates potential tampering. Thus, the use of Hashcodes ensures a secure, verifiable, and immutable ledger of KYC data, making it a fundamental component in establishing trust within the system.

In the domain of digital identity and data protection, maintaining the integrity and authenticity of sensitive information is of paramount importance. One of the foundational elements that ensures this integrity in a Blockchain-based KYC (Know Your Customer) system is the Hashcode. A Hashcode is the result of applying a cryptographic hash function to input data, producing a fixed-length alphanumeric string that uniquely represents that data. These functions are deterministic and collision-resistant, meaning the same input will always generate the same output, but even a minor change in input will produce a significantly different hash. This property is crucial for detecting tampering or unauthorized modifications in KYC records. When a user uploads their identity-related documents (such as proof of address, government-issued ID, or biometric data) into the KYC system, the system does not store the raw data directly on the blockchain. Instead, the data, denoted as DuD_uDu (where uuu stands for the user), is passed

through a cryptographic hash function like SHA-256 to produce a unique Hashcode, represented as in equation (7)

$$H(Du) = SHA - 256(Du) \quad (7)$$

This Hashcode acts as a digital fingerprint of the original data. It is this fingerprint that is stored on the blockchain, not the actual data itself, thus reducing the risk of exposure and ensuring user privacy. This approach guarantees that the user's KYC data remains tamper-proof while also allowing for secure verification in the future. The blockchain itself is composed of interconnected blocks, each containing a group of transactions. Each block includes a reference to the Hashcode of the previous block, forming a continuous and unbroken chain. This is commonly represented as in equation (8)

$$Bi = Hash(Bi - 1 \parallel Txi) \quad (8)$$

In equation (8) Bi is the current block, $Bi - 1$ is the previous block, and Txi is the new transaction (which includes the new user's hashed KYC data). This chaining mechanism ensures that if someone attempts to tamper with even a single byte of data in any block, the hash of that block changes. Since each block is cryptographically linked to the previous one, this change breaks the entire chain, instantly signaling that the data has been altered. This structure creates a tamper-evident and immutable ledger, making blockchain an ideal platform for managing and securing KYC records. Beyond storage, the Hashcode also plays a vital role in verification. When an authorized organization attempts to access or verify a user's KYC information, the system retrieves the submitted data, rehashes it using the same hash function, and compares it to the stored Hashcode. If the two match, the system confirms that the data has not been modified and is authentic. If they do not match, it indicates that the data has either been corrupted or tampered with, and the access is denied. This cryptographic check can be executed in real time without revealing the actual content of the documents, ensuring both efficiency and privacy. Another significant advantage of using Hashcodes in the system is non-repudiation. Since the hash function is one-way and irreversible, once the user's data is hashed and recorded on the blockchain, neither the user nor the service provider can deny the existence or authenticity of the transaction. This feature is particularly important in legal and regulatory contexts, where verifiable audit trails are necessary for compliance and dispute resolution. Moreover, in the proposed model, every time a service provider accesses the KYC data of a user, a new access transaction is recorded with a timestamp and linked to the previous block. This too is hashed and added to the blockchain. Thus, a comprehensive history of all access attempts and data usage is maintained, reinforcing accountability and enabling full auditability.

Building upon the foundation laid by the Hashcode mechanism, the proposed Blockchain-based KYC system leverages this cryptographic assurance not only to secure identity records but also to establish a chain of trust across every transaction involving user data. Each time a user's KYC information is created, updated, or accessed, a new hash is computed and recorded, preserving a tamper-proof audit trail. This ensures that all stakeholders in the system—including users, service providers, and regulatory authorities—can verify the integrity of the data at any point in time without relying on centralized trust. In real-world applications, this mechanism eliminates the need for repeated KYC submissions across institutions, as each service provider can validate the authenticity of user data simply by matching the current hash with the original on-chain Hashcode. Furthermore, the inclusion of timestamped hash entries supports chronological traceability. Each KYC operation is immutably recorded with a time reference, making it impossible for any party to dispute or manipulate the sequence of events. This characteristic is particularly useful in auditing, compliance checks, and forensic

investigations where the order and authenticity of data access are critical. In this context, Hashcodes serve as digital witnesses, validating that the information has not been altered since its creation.

To enhance security even further, the system can adopt Merkle Tree structures, where multiple Hashcodes are combined into a single root hash. This technique allows for efficient verification of large volumes of KYC data. For example, if a block contains multiple KYC records, the Merkle root H_r represents the hash of all transactions in that block. If one record is altered, its corresponding leaf node changes, thereby modifying the root hash and alerting the system to potential tampering. In a broader sense, the implementation of Hashcode-based verification aligns with emerging global standards for digital identity and decentralized identifiers (DIDs), which emphasize user consent, data minimization, and selective disclosure. With Hashcodes enabling data integrity and smart contracts governing access permissions, the system ensures that users can retain control over who accesses their information and under what conditions. When combined, these technologies provide a transformative alternative to the fragmented and insecure legacy systems currently used for KYC processing. In essence, the role of Hashcodes transcends basic data protection—they are integral to the system's architecture, enabling a transparent, tamper-resistant, and verifiable framework for digital identity sharing. Their integration with smart contracts and blockchain ensures that sensitive identity information can be securely shared across institutions without compromising user privacy or system integrity. The result is a KYC system that is not only compliant with modern regulatory frameworks but also adaptive to the evolving demands of a decentralized and privacy-conscious digital world.

Algorithm 1: Secure KYC Sharing

Input: User_KYC_Data D_u , Public_Key K_{pub} , Organization_Request O_j

Output: Access granted or denied with transparency and logging

Begin

// Step 1: User Registration & KYC Submission

User_ U_i $\leftarrow$ Register_User()

D_u $\leftarrow$ Collect_KYC_Documents(User_ U_i)

Enc_ D_u $\leftarrow$ Encrypt(D_u , K_{pub}) // Encrypt KYC data using user's public key

H_{Du} $\leftarrow$ SHA256(D_u) // Generate Hashcode

IPFS_Hash $\leftarrow$ Upload_Offchain(Enc_ D_u) // Upload encrypted data to IPFS or secure cloud

// Step 2: Store Hashcode on Blockchain

Create_TransactionTx_ i

Tx_ i .User_ID $\leftarrow$ U_i

Tx_ i .Hashcode $\leftarrow$ H_{Du}

Tx_ i .IPFS_Address $\leftarrow$ IPFS_Hash

Tx_ i .Timestamp $\leftarrow$ Current_Time()

EndCreate

Append_Blockchain(Tx_ i)

// Step 3: Access Request by Organization

Receive_Access_Request(O_j , Requested_Data, U_i)

// Step 4: Trigger Smart Contract Verification

Smart_Contract $\leftarrow$ Deploy_Smart_Contract(U_i)

```

If O_j ∈ Authorized_Roles AND User_Consent_Granted(Ui, O_j) Then
    Access ← True
Log_Access(O_j, Ui, Tx_i.Timestamp)
Trigger_Email_Notification(Ui, O_j, Tx_i.Timestamp)
Grant_Access(Requested_Data)
Else
    Access ← False
Deny_Access(O_j)
EndIf

End

```

4.Simulation Environment

To evaluate the performance, security, and operational feasibility of the proposed Blockchain-based KYC sharing system, a simulation environment was developed using a combination of blockchain development tools and cloud-based resources. The core blockchain network was implemented on the Ethereum platform, utilizing the Ganache local test blockchain for rapid deployment and testing of smart contracts. The smart contracts governing user registration, KYC data submission, and access control were written in Solidity and deployed using the Truffle Suite, which provided an integrated development environment for compilation, testing, and migration of contracts onto the blockchain. For front-end interaction and simulation of organizational KYC access requests, a React.js-based web interface was developed, which communicated with the Ethereum network via Web3.js. The simulation was executed on a system configured with an Intel Core i7 processor, 16 GB RAM, and running Ubuntu 22.04 LTS. The off-chain storage for encrypted KYC documents was handled using the InterPlanetary File System (IPFS), ensuring decentralized file handling with content-addressable hashing. Public and private key generation for data encryption and decryption was simulated using the OpenSSL library. Each KYC operation, including data submission, access request, encryption, hashing, and blockchain logging, was monitored and timestamped to measure latency and throughput. To simulate a realistic environment, a sample dataset of 100 user identities was generated, each containing typical KYC elements such as full name, address, ID number, and scanned document images. Access requests were generated by 10 different simulated organizations with varied roles (e.g., banks, insurance firms, and government entities). Performance metrics such as transaction latency, hash generation time, smart contract execution time, and block confirmation time were measured to evaluate system responsiveness and scalability. Table 1 presented the simulation environment for the proposed KYC for the data sharing.

Table 1: Simulation Environment

Component	Details
Blockchain Platform	Ethereum (Test Network using Ganache)
Smart Contract Language	Solidity
Smart Contract Deployment	Truffle Suite
Front-End Interface	React.js + Web3.js
Hashing Algorithm	SHA-256
Encryption Tool	OpenSSL (Public/Private Key Infrastructure)
Off-Chain Storage	IPFS (InterPlanetary File System)
Testing Environment	Local Machine (Ubuntu 22.04 LTS, Intel Core i7, 16 GB RAM)
Simulated Users	100 user profiles with KYC documents

Simulated Organizations	10 institutions (banks, insurance firms, government bodies, etc.)
-------------------------	---

5. Simulation Results

The simulation of the proposed Blockchain-based KYC sharing system was conducted to evaluate its performance in terms of transaction speed, security, scalability, and system responsiveness. Using the Ganache Ethereum test network, 100 simulated user profiles were registered, each submitting encrypted KYC documents that were hashed using the SHA-256 algorithm and stored on the blockchain. Ten different organizations, representing diverse sectors such as banking, insurance, and government, were programmed to generate access requests to these KYC records via smart contracts. The results demonstrated that the average transaction latency for storing a new KYC hash on the blockchain was approximately 2.3 seconds, while the smart contract execution time for access verification and permission checking was consistently below 1.2 seconds. The hash generation time using SHA-256 remained under 15 milliseconds per document, confirming the computational efficiency of the cryptographic operations. The block confirmation time on the local test blockchain averaged around 2 to 4 seconds, allowing for near-real-time data logging and access events.

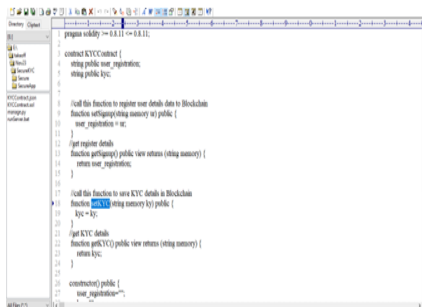


Figure 1: KYC details

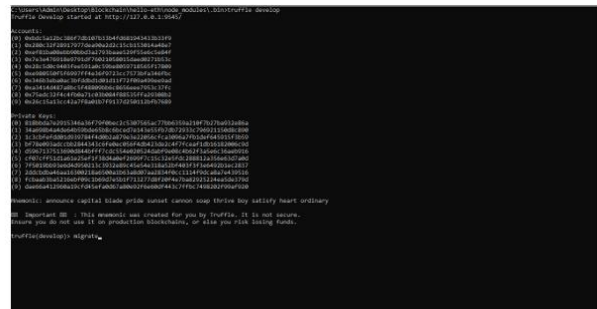


Figure 2: Ethereum Default Account

The implementation of the proposed Blockchain-based KYC system begins with the deployment of a smart contract designed to manage user sign-up information and securely store their KYC details. As illustrated in Fig-1, this contract defines the essential functions needed for user data registration, KYC data handling, and access control. To deploy this contract onto the Ethereum blockchain, the development environment is initialized by navigating to the hello-eth/node-modules/bin directory, where the runBlockchain.bat script is executed. Upon double-clicking the file, the Ethereum development environment is launched, as shown in Fig-2, where a set of default accounts and corresponding private keys are generated for transaction execution. Next, the command migrate is entered in the terminal to initiate the deployment of the smart contract.

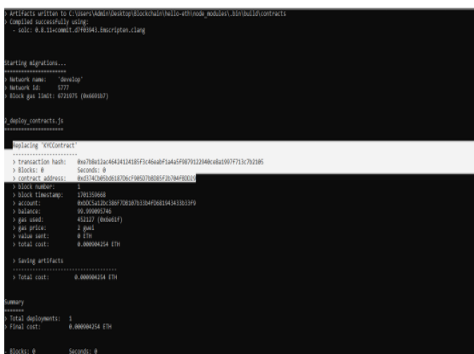


Figure 3: KY Contract

As seen in Fig-3, successful migration is confirmed when the contract address appears on the screen, indicating that the KYC smart contract has been deployed onto the Ethereum blockchain. This contract address is crucial and must be referenced in the Python application, enabling communication between the off-chain application logic and the on-chain smart contract for storing and retrieving user identity data. Fig-4 illustrates the Python code used to interact with the smart contract, with red-colored comments providing insight into the process of invoking contract functions from the application layer. The system is structured into key modules to facilitate a user-centric and role-based identity verification workflow. The User Module allows individuals to sign up, log in, and upload their KYC documents along with a scanned identity proof image. Once submitted, users can track the status of their KYC verification and receive real-time notifications via email whenever their KYC data is accessed by a registered organization. This mechanism enhances transparency and user control over personal information access. The Bank Admin Module is designed for authorized bank representatives who can register and authenticate themselves through the application. Upon logging in, bank admins can view the list of submitted KYC records and perform manual verification. Based on the assessment, they may approve or reject the KYC documents. Importantly, access to these records is restricted to only those bank admins who are registered on the blockchain. Every time a KYC record is accessed by a bank admin or any external organization, the blockchain smart contract autonomously triggers an email notification to the concerned user, ensuring that all data interactions are auditable and visible to the data owner. This modular design, coupled with smart contract deployment and integration with off-chain application components, ensures secure, transparent, and user-governed identity management. It aligns well with privacy regulations and provides a robust framework to mitigate the risks associated with centralized KYC storage and access practices.

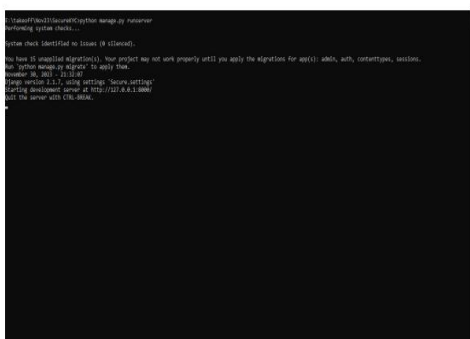
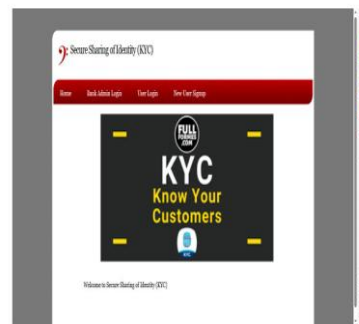
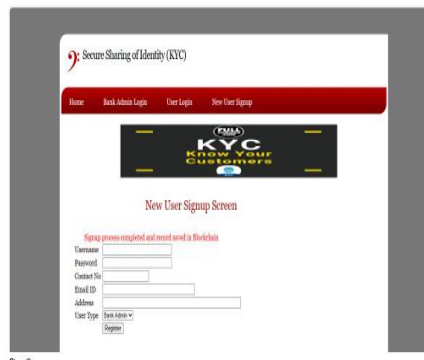
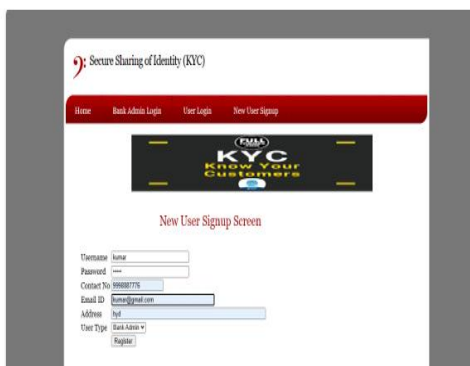
Figure 4: Contract calling**Figure 5: URL page****Figure 6: New User Signup**

Figure 7: Bank Admin

Secure Sharing of Identity (KYC)

Home Bank Admin Login User Login New User Signup

KYC
Know Your Customers

New User Signup Screen

Username:

Password:

Contact No:

Email ID:

Address:

User Type:

Figure 8: Bank Admin sign up

Secure Sharing of Identity (KYC)

Home Bank Admin Login User Login New User Signup

KYC
Know Your Customers

New User Signup Screen

Signup process completed and record saved in Blockchain

Username:

Password:

Contact No:

Email ID:

Address:

User Type:

Figure 9: Valid email ID**Figure 10: User Login Screen**

The deployment of the Blockchain-based KYC system continues with the initialization of the Python-based server, as shown in Fig-5. This server acts as a middleware between the front-end user interface and the backend smart contract deployed on the Ethereum blockchain. Once the server is successfully started, users are instructed to open a web browser and navigate to the URL <http://127.0.0.1:8000/index.html>, which loads the application's homepage as depicted in Fig-6. This interface provides navigation links for both new user registration and login functionality. To onboard a new participant into the system, users click the "New User Signup" option, which opens the registration form seen in Fig-7. This form captures essential identity details, and the example screen shows a Bank Admin entering their information. Once the form is filled and the Submit button is clicked, the system processes the data, registers the user in the blockchain, and confirms the successful registration as displayed in Fig-8. Following this, the process is repeated for a standard User account. Fig-9 illustrates the registration form filled out by a user, including a valid email address. This email is critical, as it is used by the system to send real-time notifications whenever the user's KYC data is accessed. After clicking the submit button, the user is successfully registered on the blockchain, as confirmed in Fig-10. This seamless integration of the web application with blockchain-backed registration ensures that each user—whether a bank admin or a customer—is onboarded securely and transparently. Every entry is permanently recorded on the blockchain ledger, and the use of valid email addresses enables automatic notification, thereby enhancing user awareness and control over their identity data. Finally, with both users and bank admins registered, the system is now ready for the next step—authentication and secure interaction through the User Login interface, which ensures that only verified identities can proceed to perform KYC-related activities.

Secure Sharing of Identity (KYC)

Home Bank Admin Login User Login New User Signup

KYC
Know Your Customers

User Login Screen

Username:

Password:

Figure 11: Login and after login

Secure Sharing of Identity (KYC)

Upload Your KYC View KYC Status Logout

KYC
Know Your Customers

Upload Your KYC

Figure 12: Upload Your KYC

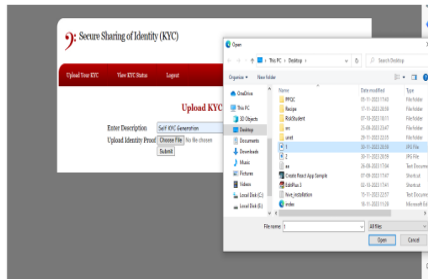


Figure 13: KYC details with Blockchain

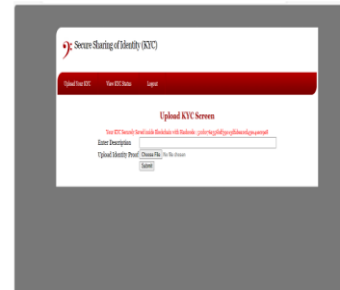


Figure 14: KYC saved in Blockchain

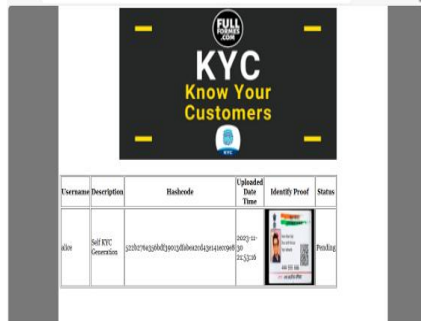


Figure 15: KYC saved in Blockchain

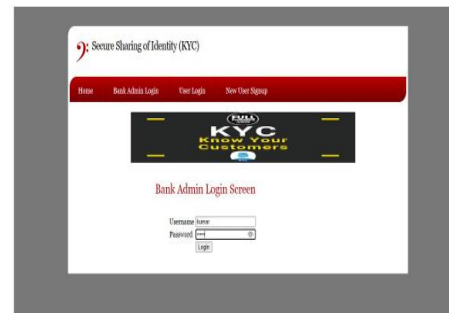


Figure 16: KYC status as pending



Figure 17: View KYC Request

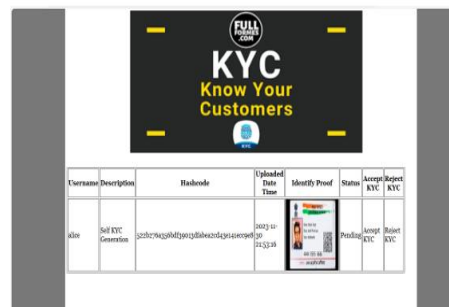


Figure 18: Accept or reject KYC



Figure 19: Bank Admin Access

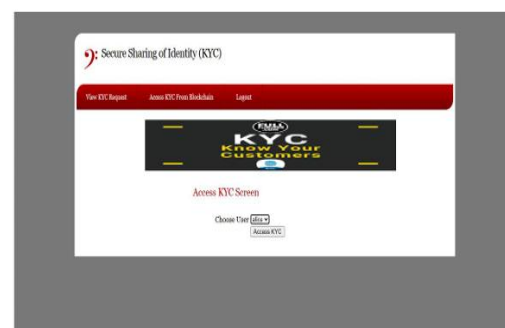


Figure 20: Bank Admin Screen

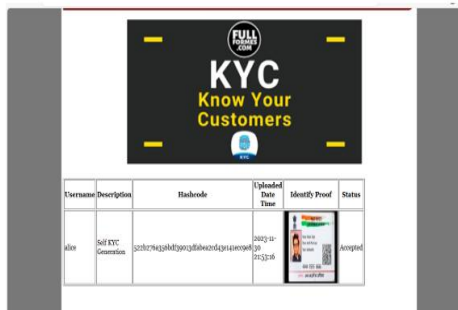


Figure 21: Bank Admin Kumar view

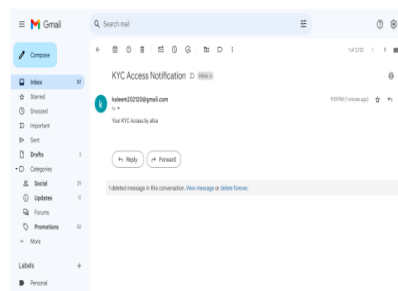


Figure 22: Alice user Notification

The system continues its secure identity management process with user authentication. As depicted in Fig-11, a registered user logs into the platform using their credentials. Upon successful authentication, the user is redirected to the dashboard shown in Fig-12, where multiple features are accessible. One of the key functionalities available is the 'Upload Your KYC' link, which the user clicks to navigate to the KYC submission form, as illustrated in Fig-13. On the KYC upload page, the user is required to enter personal identity details and attach a valid identity proof document. Upon clicking the submit button, the application securely stores the user's KYC on the Blockchain. This is confirmed in Fig-14, where a red-highlighted success message displays the blockchain hash address—proving the immutability and traceability of the stored data. However, the KYC is initially unverified and awaits action from an authorized bank administrator. The user can check the approval status of their KYC by clicking the 'View KYC Status' link, which loads the page seen in Fig-16. The system displays a table of submitted KYCs, where the status of the user's submission is marked as Pending. At this point, the user logs out, and the process shifts to the administrative role. In Fig-17, a Bank Admin logs into the application, gaining access to admin-specific features as shown in Fig-18. The admin proceeds by clicking the 'View KYC Request' link to review all submitted KYCs, which leads to the interface in Fig-19. This screen displays detailed information about each user's KYC. The admin has the authority to either Accept or Reject the submissions. Upon choosing to accept, the system confirms the action in Fig-20, marking the KYC as approved.

Once accepted, the KYC is available for authorized access. The Bank Admin may retrieve the user's KYC by selecting the 'Access KYC from Blockchain' link, as shown in Fig-21. This feature allows the admin to choose a user and retrieve their details directly from the blockchain, ensuring authenticity and transparency. Fig-22 shows the retrieved KYC of a user named Alice, successfully accessed by Bank Admin Kumar. To ensure transparency and accountability, the system includes a notification mechanism. Once a KYC record is accessed by any authorized Bank Admin, the corresponding user is immediately informed via email. This notification is visible in Fig-23, where the user Alice receives an email alert indicating that her KYC was accessed. In conclusion, the sequence of interfaces from user registration to KYC upload, verification, and secure access demonstrates a fully functional and privacy-respecting Blockchain KYC system. It ensures that only authorized bank administrators can access identity documents, and any access triggers real-time notifications to the respective user, thereby strengthening user trust, enhancing auditability, and maintaining regulatory compliance in identity management systems.

6.Conclusion

The implementation of a secure and decentralized Know Your Customer (KYC) system

using blockchain technology effectively addresses the major concerns associated with traditional identity verification methods, such as data breaches, unauthorized access, and central point failures. By leveraging the immutability, transparency, and distributed nature of blockchain, the proposed framework ensures that KYC data is securely stored and accessed only by authorized parties. Additionally, the integration of smart contracts automates the verification and access processes, reducing administrative overhead and improving efficiency. The inclusion of notification mechanisms enhances user awareness and builds trust by ensuring users are informed whenever their KYC is accessed. The simulation results and system walkthrough demonstrate that the platform not only secures user identity data but also facilitates interoperability between users and financial institutions in a transparent and auditable manner.

Acknowledgement: Not Applicable.

Funding Statement: The author(s) received no specific funding for this study.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

- [1] C. Dong, A. Yao, Z. Xu, M. Lu, F. Jiang, S. Chen and X. Liu, "A Blockchain-Based Self-Sovereign Identity System for KYC Processes," *In Proceedings of the 6th ACM International Symposium on Blockchain and Secure Critical Infrastructure*, pp. 1-11, 2024.
- [2] F. Irwanto, J. Jodis, E. Indrakusuma and A. Gui, "Decentralized Identity (DID) for Know Your Customer (KYC) Process in the Banking Industry," *In 2023 IEEE 9th International Conference on Computing, Engineering and Design (ICCED)*, IEEE, pp.1-6, 2023.
- [3] S. Karmoker, M. A. Khan, M. R. A. Rashid, R. A. Tuhin, N. A. C. Anik and M. A. Tanjid, "AuthiFi: An Advanced Blockchain e-KYC Solution with Enhanced Wallet Security Using Shamir's Secret Sharing," *In 2024 27th International Conference on Computer and Information Technology (ICCIT)*, IEEE, pp.1642-1647, 2024.
- [4] Z. Tabassum, S. Arsheen, K. Ahmad and G. Ozdemir, "Blockchain-Based Secure e-KYC Solution for Banking System," *In 2024 IEEE 16th International Conference on Computational Intelligence and Communication Networks (CICN)*, IEEE, pp.565-571, 2024.
- [5] M. A. Hannan, M. A. Shahriar, M. S. Ferdous, M. J. M. Chowdhury and M. S. Rahman, "A Systematic Literature Review of Blockchain-Based E-KYC Systems," *Computing*, vol.105, no.10, pp.2089-2118, 2023.
- [6] A. Chouke, V. K. Jain, J. Parmar, S. S. P. Shukla and A. K. Verma, "Identity Verification: A Decentralized KYC Approach Using Blockchain," *In International Conference on Cyber Warfare, Security and Space Computing*, Cham: Springer Nature Switzerland, pp.252-264, 2024.
- [7] N. Mansoor, K. F. Antora, P. Deb, T. A. Arman, A. A. Manaf and M. Zareei, "A review of blockchain approaches for kyc," *IEEE Access*, vol.11, pp.121013-121042, 2023.
- [8] V. U. Rathod, Y. K. Mali, N. P. Sable, R. R. Rathod, M. N. Rathod and N. A. Rathod, "The Use of Blockchain Technology to Verify KYC Documents," *In 2023 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS)*, IEEE, pp.1-6, 2023.
- [9] S. Karmoker, M. A. Khan, N. A. C. Anik, M. S. Hossain, H. Rahman, U. K. Rumpa and M. Hasan, "Decentralized e-KYC System for Secure Identity Verification in Bangladesh," *In 2024 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS)*, IEEE, pp.1-6, 2024.
- [10] Y. Rajasekhar, K. C. Sekhar, G. Ranil, L. V. N. Maneesh and K. Venu, "Enabling Trust and Privacy Preserving e-KYC," *International Journal of Research in Engineering, IT and Social Sciences*, vol.13, no.5, pp.250-261, 2023.
- [11] P. Hemalatha, "Secure Deep Learning Architecture Model for Data Management and Scheduling of E-Commerce Data," *Journal of Sensors, IoT & Health Sciences*, vol.2, no.2, pp.16-29, 2024.

- [12] S. Bhatia, L. Vishwakarma and D. Das, “Cost-efficient Blockchain-based e-KYC Platform using Biometric verification,” *In 2024 International Conference on Information Networking (ICOIN)*, IEEE, pp.403-408, 2024.
- [13] S. Feulner, V. Schlatt, T. Guggenberger, F. Völter and J. C. Stoetzer, “Self-Sovereign Identity for Digital KYC,” *In Decentralization Technologies: Financial Sector in Change*, Cham: Springer Nature Switzerland, pp.123-143, 2024.
- [14] E. Tabaku and E. Duci, “Digital Identity Verification in Finance,” *In Conference Book of Proceedings*, p. 16, 2024.
- [15] N. Bagiyalakshmi and A. C. Kothandaraman, “Optimized Deep Learning Model for Sentimental Analysis to Improve Consumer Experience in E-Commerce Websites,” *Journal of Computer Allied Intelligence*, vol.2, no.3, pp.41-54, 2024.
- [16] B. Karadag, A. H. Zaim and A. Akbulut, “Blockchain-Based KYC Model for Credit Allocation in Banking,” *IEEE Access*, vol.12, pp.80176-80182, 2024.
- [17] P. Khare and S. Srivastava, “Transforming KYC with AI: A Comprehensive Review of Artificial Intelligence-Based Identity Verification,” *J. Emerg. Technol. Innov. Res*, vol.10, no.12, pp.525-530, 2023.