IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 7 December 2024, accepted 7 February 2025, date of publication 17 February 2025, date of current version 21 February 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3542545

RESEARCH ARTICLE

A Comprehensive Blockchain-Based System for Educational Qualifications Management and Verification to Counter Forgery

SAID H. SAID[1,2], RAMADHANI S. SINDE[1], (Member, IEEE), EFRAIM M. KOSIA[1], AND MUSSA A. DIDA[1]

[1]School of Computational and Communication Science and Engineering, The Nelson Mandela African Institution of Science and Technology, Arusha 23311, Tanzania

[2]Department of Information Systems and Technology, The University of Dodoma, Dodoma 41218, Tanzania

Corresponding author: Said H. Said (saids@nm-aist.ac.tz)

The work of Said H. Said was supported by the University of Dodoma through the Ph.D. Scholarship.

ABSTRACT The prevalence of fake educational credentials poses a threat to the meritocratic nature of the education system and job markets. Verification of certificates to combat forgery has been a challenging endeavor due to the weaknesses of the current methods. Blockchain, capitalizing on its unique attributes, can provide an optimal solution to certification and verification problems by ensuring disintermediation, immutability, tamper-proof, efficiency, and security. Efforts to explore its potential in addressing these problems continue to gain momentum. However, the existing blockchain-based initiatives do not offer a holistic solution to the forgery problem, as they solely focus on a single education level or institution. Furthermore, these initiatives lack the essential features required to fully address this problem. This paper proposes a comprehensive blockchain-enabled system for issuing certificates from different educational levels and institutions in the country, providing a one-stop center for verifiers, such as employers, to verify all certificates a candidate possesses. As a proof of concept, a decentralized application (DApp), ElimuChain, has been developed, utilizing smart contracts and the InterPlanetary File System (IPFS). The system is deployed on the Binance Smart Chain (BSC) blockchain to evaluate its applicability in addressing the problem in the Tanzanian context. The results demonstrate that the proposed solution successfully manages the certification and verification process, and it is cost-effective, scalable, and efficient. Moreover, its performance was compared with the previous solutions in terms of latency and throughput. The comparison results show that it performs better than the counterpart for transactional operations.

INDEX TERMS Blockchain, educational certificates, ElimuChain, DApp, forgery prevention, IPFS, smart contracts.

I. INTRODUCTION

Educational certificates are highly significant, especially in this era of credentialism, where society relies heavily on credentials as an attestation to individuals' education, knowledge, skills, and competence [1], [2], [3], [4], [5]. Acquiring such credentials can potentially unlock an individual's social and economic opportunities, leading to a prosperous life [6], [7]. For instance, they are prerequisite for employment, admission to education programs, and other careers [2]. Due to their benefits, forgery has become rampant worldwide [8], [9], including in Tanzania [10], [11]. While fake certificate recipients seek social and economic benefits, producers are motivated by financial gain. Advancements in multimedia and printing technologies have improved fake certificate production, blurring the line between counterfeit and authentic certificates [12].

The black market for fraudulent certificates is vast, involving certificates from secondary school to doctoral degrees, and it is a lucrative business with multi-billion dollar

The associate editor coordinating the review of this manuscript and approving it for publication was Dominik Strzalka.

revenue [8]. Many cases of certificate fraud have been reported worldwide, including those reported in [11], [13], [14], [15], [16], and [17]. In Tanzania, for instance, this vice poses a significant challenge that requires considerable attention and action. According to [11], about 10,000 out of 400,035 public servants in Tanzania were found to have fake secondary school and teacher training certificates. These frauds come with a lot of social and economic harm. They undermine the credibility of the education system, disrupt job markets, and compromise workforce quality, the economy, and public welfare [18], [19].

To effectively prevent forgery, the verification process needs to be improved [2], [20]. Verification usually happens when graduates submit their certificates to third parties (e.g., recruiters) as part of their application for employment, education, or other opportunities. As asserted by [21], ''certificates are useful only if they can be verified.'' The recruiters must check if the certificates presented by their candidates are authentic to avoid enlisting candidates with fake certificates. Traditionally, certificates are issued physically in paper format and secured with seals, signatures, watermarks, holograms, and other security features. This practice is still dominant in most of the countries, including Tanzania [22]. Paper-based certificate verification is predominantly manual, making it time-consuming, labor-intensive, costly, bureaucratic, unreliable, and ineffective at preventing forgeries [1], [2], [22], [23], [24]. In addition, it is hard to revoke a paper-based certificate when the need arises [9], [22].

With the advent of computer technology, electronic certificates and online verification were introduced [2], [24]. In this system, educational institutions store certificate records in centralized databases and provide an application programming interface (API) for third parties, such as employers, to access the records and verify certificates remotely [24]. This approach enhances portability while reducing manual work, costs, and labor in the process [23], [24]. However, its verification process relies on the issuers' centralized systems [23]. As a result, they introduce a single point of failure, making the system vulnerable to attacks, disasters, failures, and data loss [18], [25], [26], [27], [28]. Similarly, should the institution cease to exist, the certificate records could be lost permanently [9], [24]. Furthermore, centralized systems are vulnerable to internal threats, as officials may tamper with the system to generate fake certificates [29], [30].

Electronic certificates also suffer from different digital forgery practices. According to [31], these practices involve techniques such as morphing, splicing, resampling, retouching, and copy-move forgery (CMF). CMF, as well examined in [32], [33], [34], and [35], is the most common in image and is hard to detect due to its homogeneous context [31]. These forgeries also apply to educational certificates, manipulating elements such as logos, signatures, seals, grades, holograms, names, etc. As discussed in [36], their detection techniques are divided into active and passive. Active methods are preventive; they use digital signatures and watermarking to protect content. Conversely, passive methods are detective; they identify whether an image is authentic. These include traditional forgery detection, such as pixel-based (e.g., Discrete Cosine Transform (DCT), and Principal Component Analysis (PCA)), format-based (JPEG quantization estimates), camera-based (ties an image to a particular camera features), lighting-based (detecting lighting condition inconsistency, e.g., Random Sample Consensus (RANSAC)), geometry-based (detecting geometric anomalies), and machine learning methods such as Approximate Nearest Neighbor (ANN) and Convolutional Neural Network (CNN) [34], [35], [36].

However, these methods often require digital forensics expertise, making them less accessible to general users like employers. They also need manual intervention, which can be time-consuming and error-prone. Generally, the methods discussed above are inadequate for preventing forgery [2], [20], [23].

Blockchain technology has recently emerged as a potential solution to certification, verification, and forgery problems. By leveraging its unique attributes, such as decentralization, distribution, immutability, tamper-proof, anonymity, transparency, and trustlessness, it has transformed various sectors. Following its success in cryptocurrency, there has been a surge in its applications and adoption across other domains, such as healthcare [37], [38], [39], [40], supply chain [40], [41], [42], [43], weather [43], transport [28], [44], agriculture [39], [45], e-government [46], and education [47], [48]. It is also disrupting traditional computer fields like AI [49], IoT [39], [44], [50], [51], and cloud computing [39], [44], [51], [52], [53].

Scholars and practitioners agree that, due to its unique attributes and capabilities, it can improve the issuance and verification of certificates and effectively combat forgery [9], [23], [29], [54], [55]. In the certification, verification, and forgery prevention, it is recounted to bring efficiency, reliability, security, transparency, and many other benefits. Being immutable means that once a certificate is issued on the blockchain, it cannot be altered or forged, guaranteeing its authenticity [2], [9], [56]. Through digital signatures and cryptographic hashes, blockchain provides verifiable proof of the certificate's provenance, prevents tampering, and ensures access is limited to authorized parties [57], [58]. Cryptographic techniques, such as zero-knowledge proofs, allow graduates to prove their credentials without revealing sensitive personal information, ensuring compliance with data privacy regulations [59], [60]. The decentralization and distribution of blockchain enable certificates to be stored across multiple nodes, eliminating the risk of data loss or compromise from a single point of failure while enhancing availability [9], [55], [61]. In tandem with these properties comes disintermediation (trustlessness), which decouples dependency on the issuing institutions for certificate verification [56], [61]. This offers several benefits, including simplifying the verification process, removing bureaucracy, lowering costs, and speeding

up the process [62], [63]. It also makes the verification process graduate-centric, allowing them to independently prove the authenticity of their credentials without relying on an intermediary (issuing institution) [5], [62]. The transparency property of the blockchain allows authorized participants (institutions, employers, and students) to access a clear and auditable trail of credential issuance and increases confidence in the authenticity of certifications [9], [24], [55], [61]

Its implementation can seamlessly integrate with the traditional credentialing procedures, where certificate documents, which are typically stored in Student Record Management System (SRMS), can be issued to the blockchain via a smart contract-powered DApp, where their hashes are stored immutably in a tamper-proof manner. This allows verifiers (e.g., employers), once granted access by the graduates, to query the blockchain via the DApp and instantly verify the certificate's authenticity. This process is automated by the smart contracts, simplifying the process, and reducing administrative overhead, costs, and verification time.

Several blockchain-based applications in this area have been proposed by various scholars. However, these proposals do not offer a comprehensive solution to the forgery problem due to some limitations, as outlined below. First, they are designed to work in silos by focusing on a single education level or institution, disregarding the fact that forgery affects certificates across all educational levels, and during recruitment processes, recruiters need a one-stop center to verify all certificates a candidate holds. Second, these solutions lack the essential features and functionalities required to fully address the problem

In light of the aforementioned issues, this paper proposes a comprehensive blockchain-based system to address certification, verification, and forgery challenges. It is designed to serve as a one-stop center where issuers from various education levels in the country can issue certificates, and verifiers, such as employers, can verify all certificates a candidate holds. It is tailored to address these challenges in the Tanzanian context. Therefore, it is designed in compliance with the Tanzanian educational qualifications framework and verification ecosystem by involving all key stakeholders, government authorities, and regulations associated with the certification process in the country. It is actualized by developing and deploying a DApp called ElimuChain as a proof of concept to demonstrate the viability of the solution. It utilizes smart contracts for functional logic, and the IPFS as an off-chain system for large data to overcome storage and computational limitations of the blockchain. It was deployed on the BSC blockchain for testing and evaluating its viability. The general mechanism of ElimuChain involves hashing a certificate and storing the hash (associated with the graduate's and issuer's IDs) on the blockchain while storing the actual certificate on the IPFS, allowing verifiers to query the hash from the blockchain and actual certificate from the IPFS for verification. In this context, the DApp acts as a gateway through which users interact with smart contracts, blockchain, and IPFS.

In summary, these are the main contributions of this paper:

- Contrary to the existing solutions, our solution is comprehensive, as it can be rolled out nationwide for issuing certificates from all education levels and institutions in the country, providing a one-stop center for verifiers (e.g., employers) to verify all the certificates a candidate possesses.
- Apart from issuing and verifying certificates, ElimuChain includes government regulatory authorities to manage institutions involved in the system and provides a Role-Based Access Control and Permission (RBAC-P) mechanism. This prevents misuse by unauthorized institutions, such as degree mills, and allows government oversight of certification.
- It provides on-chain certificate revocation using smart contracts, which is a more efficient mechanism than the existing approaches.
- It utilizes IPFS as an off-chain system for storing large amounts of data (e.g., certificate files) and computing hashes, providing two key benefits: allowing verifiers to preview actual certificates during verification and improving overall system performance.
- Unlike current systems, our solution enables verification of foreign awards by storing the equivalency certificates or statements on IPFS and appending their fingerprints to the blockchain, where verifiers can consult.
- Finally, this paper provides evaluation results of the system's performance, demonstrating its applicability and effectiveness in addressing certification, verification, and forgery problems, especially when rolled out nationwide to handle certificates from different education levels and institutions.

The rest of this paper is structured as follows. Section II provides background information. Section III discusses the related work. Section IV outlines the design of the proposed system. Section V delves into the implementation of the system's prototype. Section VI presents evaluation results and discussion. Finally, Section VII concludes, highlights the limitations, and sets a direction for future work.

II. BACKGROUND

A. EDUCATIONAL CERTIFICATES IN TANZANIA: ISSUANCE AND VERIFICATION

The certification ecosystem in Tanzania comprises the government, regulatory authorities, educational institutions, graduates, and recruiters (or employers). The government, through the ministry responsible for education, oversees the provision of education in the country [64]. As such, it has established various regulatory authorities to accredit and approve the issuing institutions and monitor the provision of educational qualifications at different levels, from elementary to university. The authorities include the National Examination Council of Tanzania (NECTA) for all national examinations [65], the National Council for Technical and Vocational Education and Training (NACTVET) for Technical Education Training (TET) and Vocational Education

Training (VET) colleges [66], and Tanzania Commission for Universities (TCU) for universities [67].

Certificates are issued by educational institutions or authorities across different levels. National examination qualifications, including primary, secondary, and teacher training, are issued by NECTA; VET qualifications by VETA [68]; TET qualifications or national technical awards (NTA) by technical colleges; and university qualifications by university institutions. Currently, certificates are paper-based, with security features such as holograms, seals, signatures, watermarks, and stamps to protect them against forgeries [10], [22]. These features also serve as proof of the certificate's authenticity. Issuing institutions operate in silos, each with its own certificate format and student records database [22]. These records are stored electronically in the institutions' centralized computer systems and physically in file cabinets. It is then used to retrieve information for printing and authenticating certificates. In addition to producing, maintaining, and issuing certificates, the issuing institutions are also mandated to revoke a certificate if it was issued by mistake or due to academic misconduct. The revocation process involves publishing information about a revoked certificate. However, there is no means to physically recall a revoked certificate from the holder, potentially allowing a possibility for the holder to continue using it illegally.

Some graduates obtain educational qualifications outside the country. To be usable in Tanzania, they must be converted into their equivalency with Tanzanian qualifications standards. As a result, an equivalent certificate or statement is generated and given to the graduate for use in Tanzania. This conversion ensures recognition of these qualifications by Tanzanian employers, educational institutions, and other stakeholders.

Graduates possess multiple certificates from different educational levels and institutions. During recruitment or admission process, they must submit all certificates for verification. Due to the prevalence of forgeries, recruiters (e.g., employers) must verify each certificate to ensure candidates do not present fake credentials.

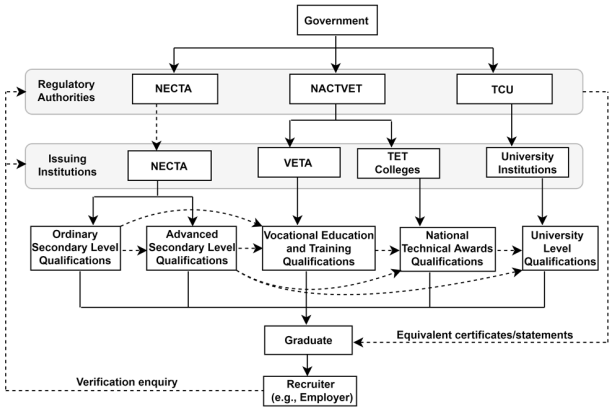

FIGURE 1. The current certification ecosystem in tanzania.

Because the certificates are paper-based, their verification is carried out manually without any sort of automation [10], [22]. Methods used, as outlined in [22] and, include consulting issuers, notarization, inspecting the security features, testing the candidate's aptitude, online verification (only for secondary certificates and few recruiters with online application systems), scanning QR codes (used by few institutions), and merely trusting a certificate by its appearance without concrete evidence. These methods are primitive, making the verification process tedious, lengthy, expensive, time-consuming, inefficient, unreliable, and ineffective in preventing forgeries. Due to these weaknesses, forgery persists, and many such incidents, especially in the employment sector, have been reported [10], [11]. Fig. 1 shows the current certification and verification ecosystem in Tanzania.

B. BLOCKCHAIN TECHNOLOGY

1) OVERVIEW

Blockchain technology is built on several technologies, including cryptography, digital signatures, peer-to-peer (P2P) networks, consensus algorithms, distributed ledgers, and programming [24], [69]. It was introduced by an anonymous entity named Satosh Nakamoto in 2008 in their Bitcoin cryptocurrency application [31]. Since then, it has evolved rapidly: Blockchain 1.0 focused on cryptocurrency, Blockchain 2.0 introduced smart contracts, Blockchain 3.0 expanded to non-financial domains via DApps, and Blockchain 4.0 has become a full-fledged business platform [2], [54], [70]. Blockchain can be defined as a digital ledger or database consisting of a continuously growing list of transactional records sealed in blocks that are chronologically chained together using cryptographic hashes to ensure their security, then distributed across multiple nodes in a P2P network [9], [55], [59], [60], [71], [72]. Its revolutionary impact lies in enabling a secure and reliable environment for strangers to transact without relying on central authorities for trust keeping [18], [73], [74], [75], instead, trust is assured by its protocols [2], [72].

In general, members of a blockchain network are identified by addresses derived from cryptographic public keys, which are generated from their private keys [59]. When users want to add a new record to the ledger, they send a cryptographically signed transaction to the network. The transaction is signed with a private key to prove its provenance, enforce non-repudiation [9], and authorize monetary expenditure on processing, payload, and smart contract execution [74]. The signed transaction is sent to the blockchain network through a node, which propagates it to other nodes until it reaches the entire network [74]. Special nodes, called miners or validators in some platforms, aggregate several valid transactions to form a block and append it to the ledger [9], [74], [75]. Validating transactions and creating new blocks involves competition among the miners or validators, using different consensus depending on the type of blockchain, and the winner creates and broadcasts the block to all other nodes [9].

A newly created block is hashed and linked to the list of previously created blocks by including a hash of the previous block in the added block, creating an immutable and tamper-proof chain of blocks [58], [71], [76]. The updated ledger is then distributed to all nodes in the network [74], [77], [78].

2) BLOCK

As shown in Fig. 2, a block comprises a header and payload (data). The header comprises metadata, such as the block's hash, hash of a predecessor block, timestamp, nonce, Merkle root, version, and others [46], [55], [59]. The timestamp specifies the time when a block was generated. The nonce is an arbitrary and a one-time integer created by a consensus algorithm and used, especially in PoW, for computing a block's hash. During the block creation competition, miners adjust this number repeatedly to find a hash that meets the set criteria, and the first one to do so is said to have mined the block [59]. The version represents the current software version of the blockchain platform. The Merkle root is a hash that represents block transactions and ensures their integrity [59]. It is obtained by recursively hashing pairs of transactions until they form a single root node at the top of the Merkle tree [59].

As stated earlier, blocks are connected by their hash values to form a chain. This ensures blockchain immutability, integrity, and tamper-proof; altering even a bit in a block changes its hash, breaking the link with the next block and invalidating the entire chain [9], [55], [72]. The chain starts with an initial block called the genesis block, which forms a foundation of all subsequent blocks. Unlike other blocks added by miners, this block is hardcoded by the blockchain platform's developers. On the other hand, the data section forms a block's payload, consisting of transactions to store in the blockchain. A transaction transfers assets, data, or values from one peer to another within a blockchain network. The number of transactions that form a block, and hence the block size, varies by blockchain platform. For example, Bitcoin's block size is limited to 1MB, while BSC and Ethereum have no fixed limit, it varies based on network activity and other factors [5].

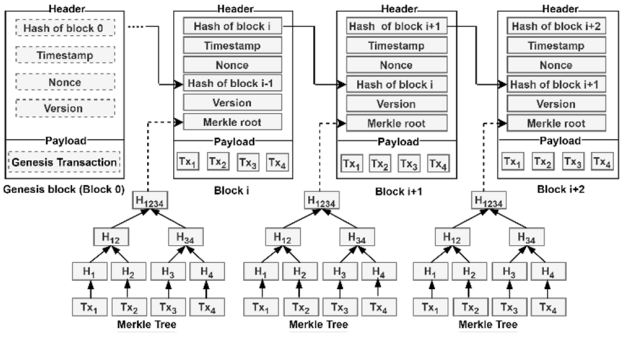

FIGURE 2. The blockchain data structure.

3) CONSENSUS ALGORITHMS

The network must agree on the validity of transactions, block, and ledger integrity [9], [74]. This is achieved through various consensus mechanisms, protocols, or algorithms. Consensus mechanisms are categorized into two: proof-based and vote-based [79]. Proof-based consensus deals with Sybil attacks [80], whereas vote-based focuses on Byzantine Fault Tolerance [81]. The most prominent proof-based consensus algorithm is Proof of Work (PoW), used in platforms like Bitcoin for its high security and integrity in decentralized and untrusted environments [63], [79]. In PoW, miners compete to solve a cryptographic puzzle of finding a specific hash for the block that meets the criteria (e.g., finding a block hash with two leading zeroes) [82]. To solve this puzzle, miners adjust the nonce and subject it iteratively to a hash function (e.g., SHA 2 or SHA 3) until they find a hash that meets the set criteria [46], [59], [63], [79], [82]. The first to find it is said to have mined the block and can append it to the blockchain. Since each iteration results in a random hash, finding a nonce that leads to the correct hash is a gamble that needs many iterations, consuming a lot of computational power [63], [79], [82]. To incentivize the miners' participation, a cryptocurrency reward is given to the winner to compensate for their computational power [9], [63], [79], [83]. Due to its high computational cost and poor scalability, alternatives like Proof of Stake (PoS) have been proposed. In PoS, validators are chosen based on their stakes (i.e., their native cryptocurrency wealth). Compared to PoW, PoS is more cost-effective and offers better performance with lower latency and higher scalability [74]. Others include Delegated Proof of Stake (DPoS), Proof of Authority (PoA), Proof of Staked Authority (PoSA), Proof of Difficulty (PoD), and Proof of Elapsed Time (PoET) [46], [74], [79]. On the other hand, vote-based algorithms rely on the vote exchange among the nodes in the network to reach a consensus, even with a certain percentage of faulty, malicious, or unreliable nodes [79]. Examples include Byzantine Fault Tolerance (BFT)-based algorithms such as Practical Byzantine Fault Tolerance (PBFT) and HoneyBadgerBFT [63], [74], [79]. BFT-based protocols are always deterministic and achieve higher performance than proof-based algorithms [79]. These protocols are typically used in permissioned and private blockchains as they require a small and predetermined number of participants [74], [79].

4) TYPES OF BLOCKCHAIN

There are many blockchain platforms that offer ecosystems to support the development of applications. Examples include BSC [84], Bitcoin [85], Ethereum [86], Hyperledger [87], Litecoin [88], Ripple [89], Holochain [90], Sovrin [91], and Corda [92].

Blockchain systems are classified as public, private, consortium, or hybrid based on ledger accessibility [46], and as permissioned, or permissionless based on the permissions to join the network [71], [79]. Public blockchains allow anyone to join, write to the ledger, read from it, validate transactions, and create blocks [2], [5], [46], [76]. They can further be classified into public permissionless or public permissioned. In public permissionless, anyone can join

and interact with the ledger without restrictions, whereas, in public permissioned, participants must be known but can interact freely with the ledger [71], [79]. On the other hand, private blockchains are owned by a single organization, which manages access and permissions [2], [5], [46]. They can also be categorized as private permissionless or private permissioned. In private permissionless, anyone can join, but only authorized participants can write to the ledger, validate transactions, and create blocks, whereas in private permissioned, both access and permission to validate transactions or write to the ledger are restricted to known participants [71], [79]. Another category is a consortium, owned by a group of organizations to provide inter-organizational services. In this blockchain, consensus is handled by the participating organizations, but reading or writing permissions may be public or limited to consortium members [2], [69], [74]. Sometimes, a blockchain may be designed by combining public and private blockchains to form a hybrid that leverages the benefits of both [46]. In this blockchain, critical information may be kept private, while others may go public [46].

5) BSC BLOCKCHAIN

This study has utilized the public-permissioned BSC blockchain. It is an Ethereum-compatible blockchain that uses its development infrastructure, such as EVM, Solidity, and others, for developing DApps [84]. It uses PoSA, which combines DPoS and PoA to achieve high throughput and fast confirmation times [84], [93]. Like Ethereum, it offers high programmability but with greater scalability, performance, and cost-effectiveness [84]. In BSC, mining is handled by validators, chosen by the network based on their stake and reputation [84]. The following is a breakdown of the process as described in [84]. (1) Once a user initiates a transaction, it is broadcast and propagated across the BSC network nodes. (2) The validators verify the transaction by checking the signature, funds, and network rules. (3) Valid transactions are placed into a pool. (4) Using the PoSA, a selected validator picks transactions from the pool, assembles them into a block, and proposes the block to the network. (5) Other validators validate the proposed block by confirming the block's transactions and adherence to protocols. (6) Once consensus is reached, the block is added to the blockchain and becomes a permanent part of the ledger.

6) SMART CONTRACTS

Smart contracts are autonomous computer programs that execute in the blockchain [59], [94]. The idea of smart contracts was proposed by Nick Szabo [95] in 1997, but it did not gain wider application until the advent of blockchain technology. Essentially, all blockchains have built-in scripts for transaction logic [71], but most, like Bitcoin, offer limited programmability [75]. Ethereum and BSC support Turing-complete and general-purpose smart contracts that can express complex logic [24], [72], [75]. In these blockchains, smart contracts are written in Solidity, compiled into EVM bytecodes, and deployed in the blockchain network. During the compilation, an Application Binary Interface (ABI), specifying how to interact with the contract, is also generated [59], [82]. Solidity-based smart contracts use the keyword "contract" (similar to "class" in Java) to define variables, functions, and other structures for data input, processing, and output. To deploy a smart contract, developers send a transaction to the blockchain containing the contract's bytecode and any necessary initialization parameters. The deployer becomes the owner of the smart contract [75]. Each smart contract is assigned a unique address on the blockchain, which is returned to the owner's wallet [59], [82]. The deployed contract can be executed by any authorized user by sending a transaction to it, specifying input data and the function to be executed [55]. The main difference between smart contracts and other programs is that smart contracts run in the blockchain autonomously, without the possibility of censorship, alteration, or interference [2], [72], [96]. Smart contracts are vital to our system, enforcing all the logic for issuance, revocation, verification, authority management, and RBAC-P.

7) INTERPLANETARY FILE SYSTEM (IPFS)

Blockchains have limited storage and computational capacity, making them unsuitable for large data and computations [55], [74], [97]. To overcome this limitation, off-chain systems are used to complement blockchain by handling large data and computations off-chain, while retaining only critical data, such as hashes on-chain [55], [74]. The IPFS, introduced by Benet in 2014 [98], is the most popular off-chain file systems used with blockchain-based systems. IPFS is a distributed content-addressable file storage and sharing system that operates on a globally decentralized P2P network, making information more resilient and accessible [20], [25], [98].

When a user adds data to IPFS, it is split into smaller chunks, hashed with SHA-256, and stored locally on each node [25], [28]. The hash serves as a Content Identifier (CID) for locating content during retrieval, making the system content-addressable [20], [25], [59]. IPFS uses a Distributed Hash Table (DHT) to keep track of the content; recording which chunk belongs to which file [98]. To retrieve content from the IPFS, users initiate a distributed lookup on the DHT to find nodes where data exists by querying the CID [28]. These blocks are then combined using a Merkle Directed Acyclic Graph (MerkleDag) to recreate the original data structure.

By integrating IPFS, the system achieves scalability and efficiency in data access and retrieval. This is achieved through its content-addressing mechanism, where files are identified by unique CIDs rather than their physical location [99]. This approach avoids duplication, reduces redundant storage, and accelerates data retrieval [100]. Being a decentralized system that distributes files across a P2P network rather than relying on centralized servers, IPFS scales

naturally as more nodes participate, eliminates single points of failure, and enhances availability and reliability [101]. Using DHT that maps file hashes to the nodes that store them, it enables quick data look up and retrieval from the network. Additionally, IPFS optimizes data access through a caching mechanism, where frequently requested content is stored locally on multiple nodes, further speeding up retrieval, minimizing latency, and ensuring availability even if some nodes go offline [100]. To further enhance efficiency, IPFS divides files into smaller chunks that can be downloaded simultaneously from multiple nodes, significantly reducing the time required to access large files [100]. To overcome the bottlenecks such as slow peer connections or limited node availability and persistent storage, IPFS uses pining services and reliable gateways (such as Infura, which is used in our system) to ensure tenacity and improve retrieval performance [100]. Unlike blockchain, IPFS does not involve consensus protocol's computational overheads to add data, as each node is responsible for only a small fraction of the content stored in the network, contributing to its high performance and scalability [101].

In our proposed system, IPFS serves as an off-chain system for actual certificate files storage and hash computations, offloading the burden from the blockchain. By storing only CIDs or metadata on-chain and keeping bulk data (e.g., certificate files) on IPFS, the system leverages both the immutability of blockchain and the scalability of IPFS, reducing on-chain storage needs and transaction costs while enhancing overall performance.

III. RELATED WORK

Several studies have explored blockchain's potential for certification and forgery prevention. Unfortunately, their proposals have limitations that prevent them from fully addressing the problem. This section reviews these works, highlighting their limitations.

Blockcerts [56] is a Bitcoin-based framework that offers tools for developers to build certificate applications [1], [5]. Students use a mobile app wallet to create a public-private key pair. While the private key is stored in their wallet, the public key is sent to the issuing institution. The issuing institution prepares certificate data in Open Badges [102], hashes it with the student's public key, and stores the hash on the blockchain. The credentials are sent to the student, who can then use the private key to independently prove ownership. In verification, students send their blockchain credentials to third parties (e.g., prospective employers), who re-compute the hash and compare it to the one stored on the blockchain. Several other projects, including Malta [103], the University of Nicosia (UNIC) [2], [57], the University of Birmingham [104], and the University of Fernando Pessoa [5], are based on the Blockcerts framework. However, Blockcerts suffer from low speed, high cost, and poor usability [2]. It lacks a mechanism for validating issuing institutions, creating a loophole for "degree mills" and impersonation of the issuer [1]. Moreover, it relies on centralized web services for off-chain revocation, raising security concerns [5].

Utilizing Ethereum blockchain, smart contracts, and IPFS, [105] developed a platform for issuing, revoking, sharing, and validating certificates to prevent counterfeits. Their solution encodes certificate information in the Open Badges, computes hash using SHA-256, and stores it on the blockchain. A certificate in JSON and PDF is sent to the learner, who can share it with third parties (e.g., prospective employers). These third parties can verify it by querying the blockchain. However, this solution cannot be universally applicable across all education levels, such as secondary to university, because it relies on a single accreditation authority.

Using ARK blockchain, [58] proposed a global higher education credit and grading system called EduCTX for issuing and verifying credits acquired by students after completing the European Credit Transfer and Accumulation System (ECTS) course. In their design, they represent ECTS credits as ECTX tokens, which are then stored in the blockchain and transferred to the student's EduCTX wallet. Students can prove ownership of the credits to third parties, such as potential employers, by presenting their blockchain address, and they can verify by querying the blockchain. The problem with EduCTX is that it is primarily designed for a single education level, i.e., higher education, particularly for institutions that follow the ECTS system, hindering its universal applicability. Moreover, it lacks a mechanism for revoking an ECTX token if an error occurs.

Another certificate system is the University of Zurich BlockChain (UZHBC), built on the Ethereum smart contracts [26]. UZHBC helps the University of Zurich issue student certificates, and third parties verify them from the blockchain. The university uploads certificates as PDF documents, hashes them using SHA-3, and stores the hashes on the blockchain via smart contracts. When verifiers want to verify a certificate, they upload it to the system, which generates its hash and compares it with the hash stored in the blockchain. However, UZHBC is tailored to the specific needs of UZH, limiting its applicability to that institution. It also lacks a certificate revocation mechanism and does not provide storage and preview of actual certificates.

Leka & Selimi [77] proposed BCert, a system that utilizes Ethereum, smart contracts, IPFS, and QR codes to store, distribute, revoke, and verify university certificates. Their solution applies a hash function on the certificate, then the resulting hash is stored on a blockchain in a transaction signed using the issuer's private key. In addition, it ensures data confidentiality by encrypting it with AES algorithm before hashing and transacting. Students receive both a certificate and a QR code to present for employer verification. It incorporates accreditation authority to certify the issuing institutions and validate certificates. However, this platform is limited to university-level certificates, making it unsuitable for managing certificates from various education levels nationwide. Additionally, it is partially decentralized as it

relies on central servers to broadcast transactions, creating a single point of failure.

Tariq et al [18] developed Cerberus, which combines a permissioned Ethereum blockchain, smart contracts, Merkle Tree, and QR code to enhance the issuance, revocation, and verification of university certificates and transcripts. Graduates' records are encoded in JSON, hashed with SHA-256, and the hashes are broadcast on the Cerberus network through a transaction signed by the issuer. They use a multi-signature scheme, involving multiple university officials in signing to enhance security. They use Merkle Tree for batch issuance to save cost and time. Graduates receive physical certificates and transcripts with QR codes that encode information stored on the blockchain. If an employer wants to verify the degree, scans the QR code with a smartphone to retrieve information from the blockchain. This solution also implements on-chain revocation via smart contracts and includes an accreditation body to administer the network, manage issuers, and control certificate issuance. Nonetheless, this solution is limited to universities, hindering its ability to scale up nationwide to address certificates from different educational levels and institutions.

VECefblock is another application developed by Nguyen et al [78] to help Vietnamese educational institutions track the student training process and store certificates in the permissioned Hyperledger Fabric, allowing enterprises to verify the records. Their solution allows teachers to submit exam results to the educational institution's local database, where a certificate is generated and sent to the blockchain using smart contracts. Graduates are then given certificate IDs that can be shared with enterprises for verifying certificates. The model was evaluated to measure its throughput and latency. Although it operates at the national level and includes certificates from all education levels, the model lacks government regulatory oversight for issuing institutions, does not have a revocation mechanism, and is partially decentralized as it uses centralized local databases for off-chain storage.

Hsu et al [2] proposed a consortium Hyperledger Fabric certificate system that enables graduates to apply for certificates, educational institutions to issue and revoke certificates, verifiers to validate certificates, and education authorities to oversee the issuance process. They introduced a facial recognition model to prevent impersonation of certificate recipients and used Hyperledger Fabric channels to implement access control. In the issuing process, recipients generate identities, including their face image, via a mobile app and send them to the issuing institution, which stores the certificate hashes, public key, and facial images on the blockchain. For verification, verifiers scan a QR code to retrieve data from the blockchain, compare it with the candidate's information, run a facial recognition check, and validate the issuers via their public key. This study lacks technical details on revocation implementation, it is partially decentralized as it relies on centralized certificate authorities (CAs) for digital identities.

Abreu et al [55] proposed a reference architecture for developing blockchain-based applications, and based on the architecture, they implemented a prototype for issuing and verifying higher education certificates in Brazil. Their design incorporates a government entity for registering issuing institutions. This work is designed specifically for higher education in Brazil. Therefore, it cannot serve as a comprehensive solution for verifying all certificates. Besides, it does not address certificate revocation.

Utilizing Ethereum smart contracts, Palma et al [63] introduced a system for Brazilian undergraduates that registers students' courses and learning history, tracks their credits, and automatically issues certificates to the blockchain for those who have fulfilled the requirements. To comply with Brazilian regulations, they have incorporated a government regulatory agency (RA) to manage higher education institutions' (HEIs) identities and approve courses. HEIs are identified using their public key certificates obtained from the Brazilian Government Public Key Infrastructure (ICP-Brasil). In its process, the RA deploys smart contracts and registers HEIs. HEIs then store students' learning records on these contracts, which track the records and automatically generate certificates upon course completion. The verifiers can query smart contracts to verify certificates. This study focuses only on higher education institutions in Brazil. It does not encompass all individual's academic records. Additionally, it lacks an alternative storage solution for large data that cannot fit on the blockchain.

Xie et al [24] proposed a certificate system for college students' innovation and entrepreneurship competitions, utilizing Ethereum and smart contracts to manage authority and handle certificate issuance, revocation, and verification. The authority management enables distinct permissions and access rights to users (system manager, issuing agency, and issuer). An administrator approves certificate-issuing agencies, which approve issuers among their employees, and the issuers create a certificate template in JSON, sign it, and submit it to the contracts for storage. A graduate receives a certificate, along with its hash, which can be shared with verifiers, and the verifier can invoke smart contracts to verify the certificate. They implement on-chain revocation by maintaining a certificate revocation list on the smart contracts. However, the study is limited to a single college and program.

Saleh et al [62] proposed a Decentralized Control Verification Privacy-Centered (DCVPC) protocol based on Hypeledger Fabric and chaincodes for issuing university certificates globally. In their proposal, governments, through their ministries, can join the blockchain and register their respective universities to issue certificates to their students. Their model preserves certificate security and privacy by providing authority management, establishing private channels for universities, and controlling access to the ledger. However, their proposal lacks a certificate revocation and is limited to a single education level, making it unable to handle all certificates a graduate may obtain from various levels.

Based on Ethereum smart contracts, Wahi et al [9] introduced CryptoCert DApp. It improves upon previous studies by storing the entire certificate on-chain using JSON,

TABLE 1. Related work compared to our proposed system, ElimuChain.

Work/Author	Technology utilized			Contribution/Functionality/Features						
	BC	SC	IPFS	I	V	R	C	MA	RBAC-P	EC
Blockcerts [56]	Bitcoin	No	No	Yes	Yes	Yes (Off)	No	No	No	No
UNIC [57]	Bitcoin	No	-	Yes	Yes	Yes (Off)	No	No	-	No
Gräther *et al* [105]	Ethereum	Yes	Yes	Yes	Yes	Yes (On)	No	Yes	Yes	No
EduCTX [58]	ARK	No	No	Yes	Yes	No	No	No	No	No
BCert [77]	Ethereum	Yes	Yes	Yes	Yes	Yes (On)	No	Yes	Yes	No
Gresch *et al* [26]	Ethereum	Yes	No	Yes	Yes	No	No	No	No	No
Cerberus [18]	Ethereum	Yes	No	Yes	Yes	Yes (On)	No	Yes	Yes	No
VECefblock [78]	Hyperledger	Yes	No	Yes	Yes	No	Yes	No	Yes	No
Hsu *et al* [2]	Hyperledger	No	No	Yes	Yes	Yes (-)	No	Yes	Yes	No
Educ-Dapp [55]	Ethereum	Yes	No	Yes	Yes	No	No	Yes	Yes	No
Palma *et al* [63]	Ethereum	Yes	No	Yes	Yes	Yes (On)	No	Yes	Yes	No
Xie *et al* [24]	Ethereum	Yes	No	Yes	Yes	Yes (On)	No	Yes	Yes	No
CertEdu [5]	Bitcon/Ethereum	No	No	Yes	Yes	Yes (Off)	No	Yes	No	No
DCVPC [62]	Hyperledger	Yes	No	Yes	Yes	No	No	Yes	Yes	No
CryptoCert [9]	Ethereum	Yes	No	Yes	Yes	No	No	Yes	-	No
Serranito [61]	Ethereum	Yes	Yes	Yes	Yes	Yes (-)	No	Yes	No	No
ElimuChain (Our Solution)	BSC	Yes	Yes	Yes	Yes	Yes (On)	Yes	Yes	Yes	Yes

Key: BC = Blockchain, SC = Smart contract, IPFS = InterPlanetary File System, I = Issuance, V = Verification, R = Revocation, C = Comprehensive, i.e., a one-stop-center that handles certificates from different educational levels and institutions, MA = Manage the authority of entities involved in the system, RBAC-P = Role-Based Access Control and Permission, EC = Handles Equivalent certificates (foreign certificates), On = On-chain, Off = Off-chain, No = Not fulfilled, Yes = Fulfilled, - = Not specified.

incorporating a central authority to approve issuers, and making it universal rather than limited to a specific institution. However, their prototype lacks certificate revocation and RBAC-P mechanisms. Though claimed to be universal, it lacks a mechanism to handle multiple certificates for a graduate obtained from various education levels and institutions.

Serranito [61] utilized Ethereum smart contracts to develop a system for issuing and verifying university certificates. However, the solution is not comprehensive to handle certificates across educational levels, as it was designed specifically for universities. It does not include a revocation mechanism and does not incorporate RBAC-P to ensure that only authorized users can access the system.

In summary, these studies have limitations that prevent them from offering a comprehensive solution to the problem. Since graduates hold multiple certificates from various educational levels and institutions, recruiters need a single platform to effectively verify all certificates. However, as shown in Table 1, most solutions, except VECefblock [78], are designed for a single educational level or institution, limiting their ability to handle all certificates a graduate may possess. In contrast, our solution offers a one-stop center for validating all certificates a graduate holds. Additionally, these studies lack the essential features needed to address certification problems fully. For example, as shown in Table 1, none of the solutions consider equivalent certificates, and only Gräther et al [105] and Leka & Selimi [77] include a decentralized off-chain system for large amounts of data and computations. Our solution integrates government regulatory authorities to manage institutions, provides RBAC-P, offers on-chain revocation, utilizes a decentralized off-chain system, and considers equivalent certificates. Existing solutions lack some of these features; as a result, they partially address the problem.

IV. THE PROPOSED SYSTEM: ELIMUCHAIN

A. SYSTEM OVERVIEW

ElimuChain enhances the issuance, revocation, and verification of educational qualifications using blockchain, smart contracts, and IPFS, enabling easy verification of credentials to prevent forgery. It advances upon previous studies by offering a comprehensive solution for issuing certificates from different educational levels and institutions in the country, providing a one-stop center for verifiers, such as employers, to verify all certificates a candidate possesses. It also incorporates government authorities to manage the issuing institutions, provides RBAC-P, and enables on-chain revocation via smart contracts. It is designed in compliance with the Tanzanian educational qualifications framework and verification ecosystem by engaging key stakeholders, government authorities, and relevant certification regulations. Tanzania has been selected as a case study to test the system's applicability, but it can be adapted to other countries or education systems. It focuses on secondary, teachers' training, VET, TET, and university certificates, as these are the prerequisites for recruitment or admission to education programs and, therefore, most prone to forgery.

B. SYSTEM ARCHITECTURE

As depicted in Fig. 3, system architecture comprises five components: Users, DApp front-end, Access, Blockchain and Smart Contracts, and IPFS.

Users: They include the government entity, regulatory authorities, educational institutions, graduates, and recruiters or employers. (i) The *government entity*, represented by the ministry responsible for education, acts as the central authority that oversees certificate issuance. This entity is responsible for initiating smart contracts and registering regulatory authorities on the blockchain. Deploying smart contracts on the blockchain generates an account for the

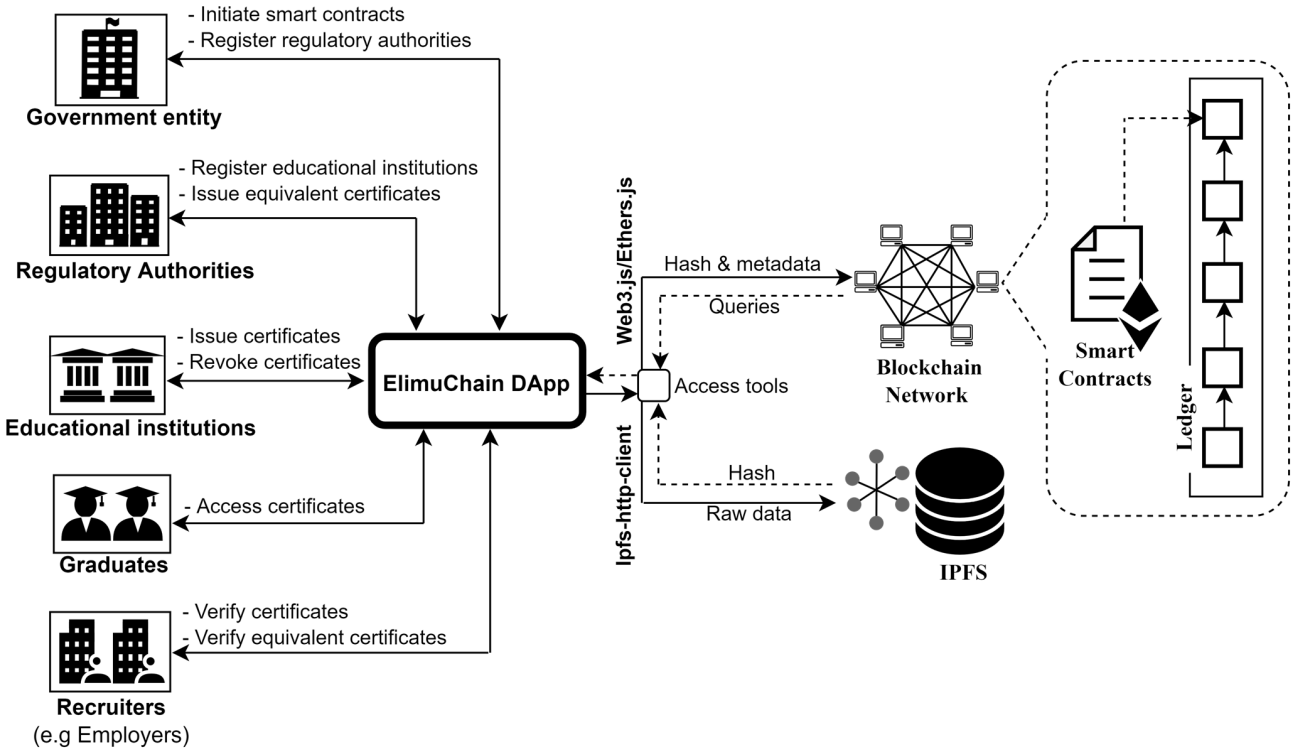

FIGURE 3. Architecture of the proposed system.

entity automatically, allowing it to access the system and perform the registration. (ii) The *regulatory authorities'* role is to register educational institutions on the blockchain and permit them to issue certificates. The system also allows the regulatory authorities to deregister or suspend the issuing institutions from the blockchain if they fail to comply with standards. Since each educational level has its own regulatory authority, the system incorporates all of them, and through smart contracts, they are allowed to register their respective education institutions. The involvement of regulatory authorities ensures that only authorized institutions can join the blockchain to issue certificates. (iii) After joining the blockchain, *educational institutions* can issue and revoke certificates. To issue, they generate a certificate in PDF, upload it to the system along with other metadata, gets hashed in the IPFS, the generated hash is stored in the blockchain while the actual certificate is stored in the IPFS. The digital certificate and its ID are then issued to the graduate. Each stored hash is associated with the graduate's and issuer's ID using smart contracts. (iv) *Graduates* are registered by their educational institutions to join the platform and access their digital certificates with their IDs. They can also receive them through other communication channels. (v) *Recruiters (employers, or verifiers)* entails any organization, enterprise, educational institutions, or others where graduates may apply for employment, further education, or other positions. After receiving the credentials from the candidates, they verify them by querying the blockchain and IPFS via the DApp. Verifiers do not need an account, as verification is open to everyone and does not involve a transactions. The rest of the users must be registered and identified using their blockchain addresses, which are created from cryptographic public keys derived from their corresponding private keys and managed by a MetaMask wallet.

ElimuChain DApp: This is an application through which users interact with the blockchain and IPFS when discharging their roles. It is powered by smart contracts, which are deployed on the blockchain and reflected to the DApp's front-end to manage all the user activities, such as transactions and queries. It is a web-based application, accessible via a web browser with the MetaMask wallet extension, and runs on various devices such as smartphones, tablets, laptops, and desktops.

Access tools: It facilitates the connection and interaction between the system's front-end and back-end. It comprises JavaScript libraries like Web3.js and Ethers.js to enable front-end scripts to communicate with smart contracts via their ABI, and Geth (Go-Ethereum) as a client implementation to connect to blockchain network nodes. It also has other services like Infura and ipfs-client to facilitate connection with both blockchain and IPFS.

Blockchain and Smart Contracts: The blockchain stores certificate hashes and their associated metadata, which are used by third parties (e.g., employers) for verifying the certificates. We have utilized public-permissioned BSC blockchain, as it can enable RBAC-P and the government oversight on the certificate issuance. It fits our use case because certificates are issued only by authorized institutions while the verification

is open to any third party. The smart contracts, deployed in the blockchain, are the main engine of the application, implementing system's functionalities, such as certificate issuance, revocation, verification, authority management, and others.

IPFS: To overcome blockchain's storage and computational limitations, the proposed system uses IPFS as an off-chain system for hash computations and storing actual certificate files. Shifting large amounts of data and computations off-chain improves the overall performance, ensures scalability, and reduces on-chain overhead while maintaining data integrity by anchoring IPFS file hashes on the blockchain.

V. IMPLEMENTATION OF THE PROPOSED SYSTEM

A. TOOLS AND TECHNOLOGIES USED

The back-end consists of smart contracts implemented using the Solidity programming language. Remix IDE [106] and Visual Studio Code [107] were used for implementing the Solidity contracts. Remix is an open-source Integrated Development Environment (IDE) that provides a user-friendly interface for writing, deploying, and testing smart contracts on the blockchain network, whereas, Visual Studio Code is a full-stack code editor for both front-end and back-end. The contracts were compiled using the Solidity compiler (solc), to convert them into bytecodes, which are interpreted by the EVM, and then deployed in the blockchain. Although smart contracts reside on the blockchain, they are mirrored to the front-end via the ABI, enabling communication between the two.

The front-end of the ElimuChain was built using React.js [108], a JavaScript library comprising HTML, CSS, and JavaScrip for building user interfaces that can interact with the blockchain. React.js has enabled us to run and manage our DApp front-end via Node.js [109], a JavaScript runtime environment that provides front-end server.

To connect the ElimuChain front-end to the blockchain nodes (DApp's back-end), Web3.js [110] and Ethers.js [111] libraries were used. Web3.js and Ethers.js are JavaScript libraries that provide APIs for interacting with the blockchain, enabling tasks like connecting to nodes, sending transactions, interacting with smart contracts, and retrieving data from the blockchain.

They receive instructions and rules from ABI in JSON, on how to interact with the smart contracts. MetaMask wallet [112], a browser extension for Chrome and Firefox, was used to facilitate interaction with the blockchain from the web browser, as well as to store and manage accounts, blockchain credentials, and cryptocurrency. For client implementation, we have utilized Go-ethereum (Geth) [113], a popular Ethereum client software, which is also applicable in BSC. It runs in the network nodes, allowing computers to connect to the network, synchronize with the blockchain, maintain a local copy of a network state, and participate in the consensus process. Connection with IPFS was achieved using Infura [114], a service that provides remote access to blockchain and IPFS networks without the need to set and maintain a node. IPFS node integration with the DApp over HTTP was achieved via ipfs-http-client [115].

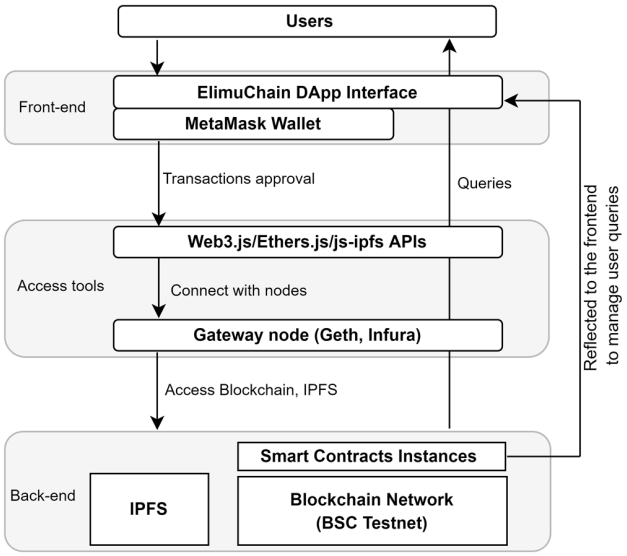

FIGURE 4. The interplay of the used tools (technologies, libraries, and frameworks).

The DApp was tested and evaluated on the BSC Testnet [116], as it provides a closer simulation of the BSC mainnet. Table 2 describes the tools (technologies, libraries, and frameworks) utilized during the implementation and experimental evaluation. Fig. 4 shows an interplay of these tools in the system's implementation and evaluation experimental setup.

It shows that users initiate and approve transactions via the MetaMask wallet, while the access tools connect and intermediate communication with the back-end. The smart contracts deployed in the back-end are reflected back in the DApp's frond-end to manage user requests. Querying information from the blockchain is not a transactional operation, hence it does not involve the wallet. The source code is available at https://github.com/saidhsaid/ElimuChain.

B. SMART CONTRACTS IMPLEMENTATION

The system's business logic was implemented using three smart contracts: *Institution*, *Certification, and AccessControl.* While *Institution* and *Certification* contracts were custom-developed, *AccessControl* was inherited from OpenZeppelin's Solidity libraries.

By inheriting *AccessControl,* we were able to utilize its key features, *_setupRole()* and *hasRole()*, to implement Role-Based Access Control and Permissions (RBAC-P), allowing the definition of various roles and assignment of specific permissions. This helps in enhancing the security and integrity of the DApp by ensuring that only authorized entities can perform designated actions within the system.

As previously mentioned, these contracts are deployed on the blockchain by the government entity, which acts as the owner of the contracts. Fig. 5 illustrates the architecture of

TABLE 2. Tools (Technologies, libraries, and frameworks) utilized in the implementation of ElimuChain DApp.

Tools, Technologies	Version	Description
BSC	-	A blockchain platform, used for deploying and testing the ElimuChain DApp prototype
IPFS	0.24.0	Decentralized, distributed, and immutable off-chain system for hash computation and certificates file storage
Solidity	^0.8.0	Ethereum's high-level object-oriented programming language, used for implementing the smart contracts
Remix IDE	0.37.3	IDE for writing, deploying, and testing Solidity smart contracts on the blockchain
MetaMask	11.5.1	A wallet (browser plugin), to manage user accounts i.e. passwords, keys, and addresses, but also transactions and fees
Web3.js/Ethers.js	4.2.2/6.8.2	JavaScript library or API for connecting DApp's front-end to the blockchain nodes or DApp's back-end
React.js	18.2.0	JavaScrip library for developing a front-end that can interact with blockchain. It includes HTML, CSS, and JavaScript
Node.js	20.9.0	A server-side JavaScript runtime environment for writing, running, and managing the DApp front-end
Go-ethereum	1.13.5	An official Ethereum client software implementation, used for running Ethereum or BSC nodes
Infura	-	Service that connects the DApp with a node, which is the gateway to the blockchain network and IPFS network
BSC-Testnet	-	Network for testing and evaluating the performance of the BSC-based DApps
Visual Studio Code	1.84	A full-stack code editor for implementing both the front-end and back-end of the DApp

these contracts, highlighting their attributes, functions, and their relationships.

1) THE INSTITUTION SMART CONTRACT

This contract manages the authority of the entities involved in the system and ensures role-based access control and permissions. It allows the government entity (central authority) to register the regulatory authorities and grant them permission to register their respective educational institutions. After its deployment, an instance or object of the contract is created on the blockchain and assigned a unique address. This address enables authorized entities to interact with the contract and use its functions for various purposes. It uses two structs (data structure): *struct Regulator (see* Fig. 6) and *struct Institute* (see Fig. 7).

The *Regulator* struct represents and stores information about a regulatory authority within the smart contract, encapsulating its properties and permissions. This struct includes the following attributes:

- *ipfs_hash*: An IPFS hash pointing to regulator-related data stored on the IPFS network.
- *regulator_address*: The blockchain address of the regulator.
- *regulator_name*: The regulator's name, acronym, or identifier.
- *canGenerateCertificate*: A boolean indicating whether the regulator has permission to generate certificates.
- *canGenerateEquivalentCertificate*: A boolean showing whether the regulator can issue equivalent certificates.

The last two variables are particularly significant in the Tanzanian context, where equivalent certificates are typically issued by regulatory authorities, and some of these authorities serve dual roles as both issuers and regulators. When an instance of this struct is created, it is stored in its *mapping* data structure (see Fig. 6, line 8) and executed within the smart contract functions to keep track of the regulators and their attributes.

The *Institute* struct represents and stores information about an educational institution in the smart contract, including its properties and permissions. This struct includes the following attributes:

- *ipfs_hash*: An IPFS hash representing institution-related data stored on IPFS network.
- *institute_address*: The blockchain address of the institution.
- *regulator_address*: The blockchain address of the regulatory authority where the issuer belongs.
- *institute_name*: The institution's name, acronym, or identifier.
- *canGenerateCertificate*: A boolean indicating whether the institution has permission to generate certificates.
- *status*: A boolean indicating the registration status of educational institution.

The latter variable allows regulatory authorities to suspend or dis-accredit educational institutions for non-compliance, such as failing to meet quality standards. When an instance of this *struct* is created, it is stored in its *mapping* data structure (see Fig. 7, line 9) and executed within functions of the smart contract to keep track of the issuing institutions and their attributes.

The *Institution* contract includes several functions, as indicated in Fig. 5. Among them, *addRegulator()*, and *addInstitution()* are the most significant.

The *addRegulator()* function adds a new regulatory authority to the smart contract, ensuring access control, checking for duplicates, assigning roles, and emitting an event to notify external entities about the addition. Algorithm 1 describes the function, showing the process of adding a regulatory authority to the system. It receives the regulatory authority's attributes as input and outputs a message indicating whether the transaction was successful or unsuccessful. This function can only be invoked by the Government entity (contract owner) through the interface in Fig. 9, as it is the one responsible for adding the regulatory authority.

The algorithm starts by authenticating the caller in lines 2-4. Then, in lines 6-13, it iterates through an array of existing regulatory authorities stored in a *mapping* data structure (see Fig. 6, line 8) to check for duplicates (if the regulatory authority to be added already exists). If there are no duplicates, a new instance of the *Regulator* struct is created with the provided information and appended to the array of regulators in the *mapping* (see line 14). The *mapping* associates the regulatory authority's ID with an instance of the

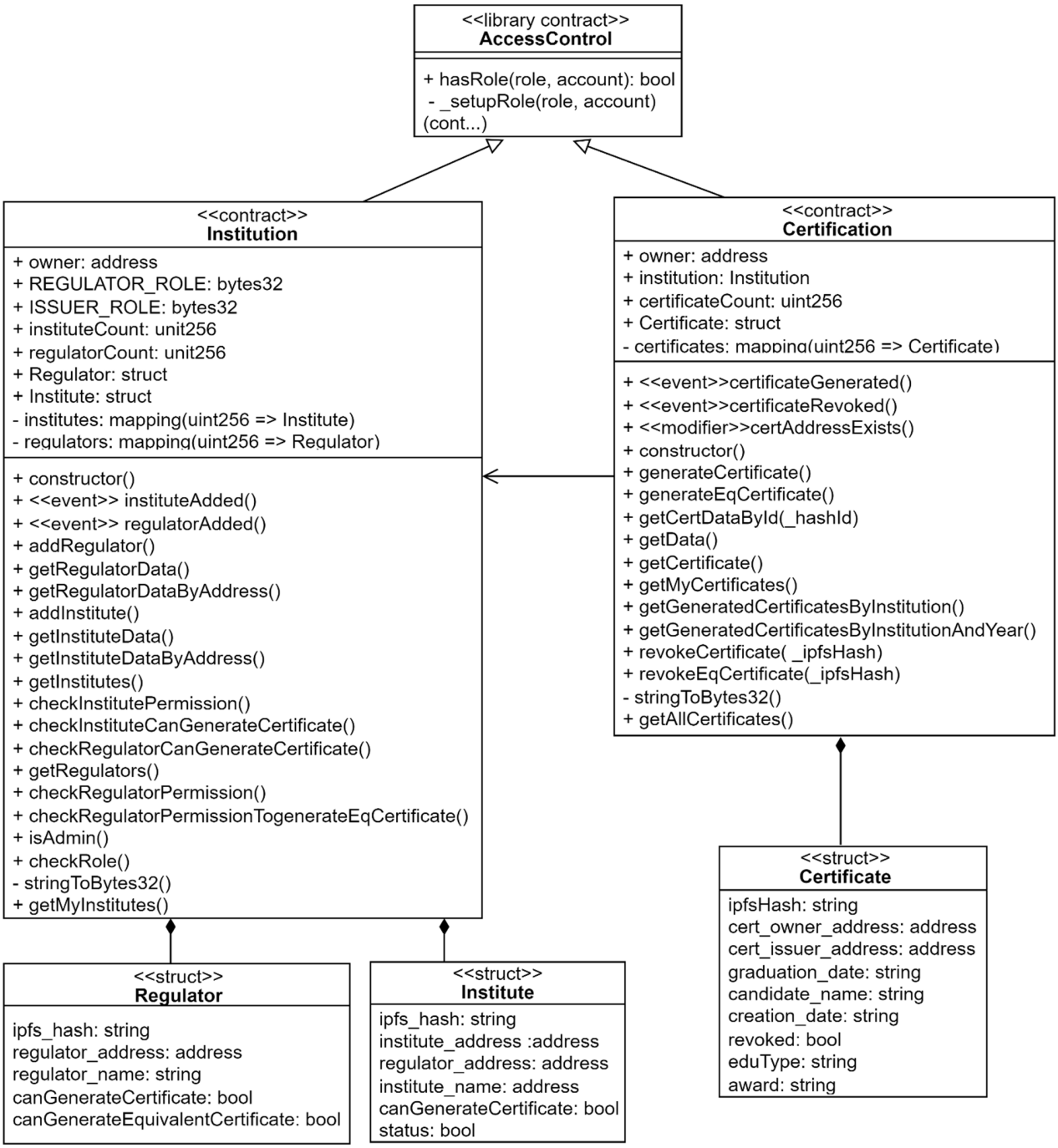

FIGURE 5. The employed smart contracts: attributes, functions, and relationship.

Regulator struct. Then, in line 15, a *regulator_role* is assigned to the newly added regulator, mandating its regulatory authority capability. It then emits an event, in line 17, indicating that a new regulatory authority is added.

The *addInstitute()* function adds a new educational institution to the smart contract, enforcing RBAC-P, checking for duplicates, assigning roles, and emitting an event to notify external entities about the addition. Algorithm 2 outlines the function, showing the process of adding an educational institution to the system. It receives the educational institution's details as input and outputs a message indicating whether the transaction was successful or unsuccessful. Since educational institutions must be registered by their respective regulatory authority, this function can only be called by the regulatory authority responsible for the educational institution being added, through an interface in Fig. 10. The algorithm begins by checking if the caller is an authorized regulator and has a regulatory role (see lines 2-4). Then, in lines 6-13, iterates

Algorithm 1 Adding regulatory authority to the system

Input: _address ra, _ipfs_hash h, _regulator_name rn, _canGenerateCertificate gc, _canGenerateEquivalentCertificate ge, list of regulators regulators[i], regulator_role from _setupRole(), regulatorCount, event regulatorAdded()
Output: true if regulatory authority added successfully, false otherwise

```
Begin
    if(msg.sender != owner) then
        revert: ''the caller must be the owner of contract, i.e., government entity''

    end if
    temp ← 0 // initialize temporary regulator variable
    for i ← 0 to regulatorCount - 1 do //loop through existing regulators to check if a newly added alreadyexists
        if (regulators[i].regulator_address = _address) then
            temp ← regulators[i] // if the regulator exists in mapping store its details in temp variable
        end if
    end for
    if (temp is not empty) then
        revert: ''the regulator with this address already exists''
    end if
    regulators[regulatorCount] ← Regulator(ra, h, rn, c, e) // create a new Regulator struct and add to the array of
    regulators
    _setupRole(regulator_role, ra) // assign regulator_role to the new regulator
    regulatorCount ← regulatorCount + 1 // increment the regulators count
    emit regulatorAdded(h, n) // emit event indicating a new regulator is added
    return true // transaction successful
End
```

```
1.  struct Regulator {
2.          string ipfs_hash;
3.          address regulator_address;
4.          string regulator_name;
5.          bool canGenerateCertificate;
6.          bool canGenerateEquivalentCertificate;
7.  }
8.  mapping(uint256 => Regulator) private regulators;
```

FIGURE 6. Regulatory authority data structure and its mapping.

```
1.  struct Institute {
2.          string ipfs_hash;
3.          address institute_address;
4.          address regulator_address;
5.          string institute_name;
6.          bool canGenerateCertificate;
7.          bool status;
8.  }
9.  mapping(uint256 => Institute) private institutes;
```

FIGURE 7. Educational institution data structure and its mapping.

through an array of the existing educational institutions stored in a *mapping* data structure (see in Fig. 7, line 9) to check for duplicates (if the educational institution being added already exists). If there are no duplicates, a new instance of an *Institute* struct is created with the given information and appended to the array of institutions in the *mapping* (see line 14). The *mapping* associates the educational institution's ID with an instance (object) of the *Institute* struct. Note: The function ensures that a regulatory authority only deals with the educational institutions under its jurisdiction, reducing the risk of misuse or tampering. Then, in line 15, an *issuer_role* is assigned to the newly added educational institution, mandating its certificate-issuing capability. It then emits an event, in line 17, indicating that a new educational institution is added.

2) THE CERTIFICATION SMART CONTRACT

This contract is responsible for the core functions of the system, including issuance, revocation, and verification of certificates and equivalent certificates. After deployment, its instance is formed on the blockchain and assigned a unique address. This address allows authorized entities to call the contract and interact with its functions for various purposes. The certificate data are stored in a data structure, *struct Certificate* (see Fig. 8, lines 1-11), which defines several attributes to encapsulate certificate details. The defined attributes include:

- *ipfs_hash*: An IPFS hash referencing an actual certificate stored on the IPFS network.
- *cert_owner_address*: The graduate's blockchain address.
- *cert_issuer_address*: The issuing institution's blockchain address.
- *graduation_date*: Sets the date when the student graduated.
- *candidate_name*: The name of the certificate holder or graduate.
- *creation_date*: A timestamp or the date when the certificate was issued to the blockchain.

Algorithm 2 Adding educational institutions to the system

Input: _address ia, _institute_name in, _regulator_address ra, _ipfs_hash h, _canGenerateCertificate gc, _status s, _canGenerateEquivalentCertificate e, list of regulators institutes[i], issuer_role from _setupRole(), issuerCount, event issuerAdded()
Output: true if educational institution is added successfully, false otherwise

```
1  Begin
2      if not (checkRegulatorPermission(msg.sender) && hasRole(regulator_role, msg.sender)) then
3          revert: "caller must be authorized regulatory authority"
4      end if
5      temp ← 0 // initialize temporary institute variable temp
6      for i ← 0 to instituteCount − 1 do // loop through existing institutions to check if newly added already exists
7          if (institutes[i].institute_address = msg.sender) then //
           check if the education inst associated with the RA exists in mapping
8              temp ← institutes[i] // store its details in temp variable
9          end if
10     end for
11     if(temp is not empty) then
12         revert: "the educational institution already exists"
13     end if
14     institutes[instituteCount] ← Institute(h, ia, ra, in, gc, s) // create a new Institute struct and add to the array of edu
       institutes
15     _setupRole(issuer_role, ia) // assign issuer_role to the newly added edu institution
16     instituteCount ← instituteCount + 1 // increment the issuer count
17     emit issuerAdded(h, in) // emit event indicating a new issuer is added
18     return true // transaction successful
19 End
```

- *revoked*: A flag indicating certificate revocation status (true if revoked, false otherwise)
- *eduType*: Differentiates between a normal certificate and an equivalency certificate.
- *award*: Describes an award.

Additionally, this contract defines certificate *mapping* (see Fig. 8, line 12) that pairs a certificate ID with an instance (object) of the *Certificate* struct. When the Certificate's instance is created, it is stored in the *mapping*, ready to be manipulated by the contract's functions.

The *Certification* contract includes several functional and utility functions, as shown in Fig. 5. Among them, the key functions are: *generateCertificate()*, *revokeCertificate()*, *getCertDataById()*, *generateEqCertificate()*, *revokeEqCertificate()*, and *getMyCertificates* These functions implement the core functionalities of the system, making them essential to its operations.

The *generateCertificate()* function handles the certificate-issuing process to the blockchain. To issue a certificate, registered educational institutions upload certificate information and actual certificate files on the DApp via the interface in Fig. 11. The DApp sends the PDF file to the IPFS through an Infura gateway, where it is hashed with the hash function, SHA-256, to generate a unique hash. The certificate file is stored on the IPFS, while its hash is sent back to the DApp, which then sends it to the blockchain. After signing a transaction with the issuer's private key using the wallet,

```
1.  struct Certificate {
2.      string ipfs_hash;
3.      address cert_owner_address;
4.      address cert_issuer_address;
5.      string graduation_date;
6.      string candidate_name;
7.      string creation_date;
8.      bool revoked;
9.      string eduType;
10.     string award;
11. }
12. mapping(uint256 => Certificate) private certificates;
```

FIGURE 8. Certificate data structure and its mapping.

the DApp invokes the *generateCertificate()* function to issue a certificate. Algorithm 3 shows the function's execution process in issuing a certificate to the blockchain. It receives the hash value and other certificate details as parameters. Since only authorized educational institutions can issue certificates to the system, the execution flow starts by verifying the permissions of the institution attempting to add the certificate (in lines 2-4). This is achieved by calling the *checkInstitutePermission()* function from the *Institution* contract and passing the caller's blockchain address to it. This process ensures that the transaction originates from an authorized educational institution, preventing system misuse by fake institutions. Then, in lines 6-14, it traverses the certificates *mapping* data structure (see Fig. 8, line 12) to check if the certificate hash being added already exists to avoid duplicates. If it does not

Algorithm 3 Issuing a certificate to the blockchain

Input: ipfs_hash ch, cert_owner_address sa, cert_issuer_address ia, graduation_date gd, candidate_name sn, creation_date cd, revoked r, eduType et, year y, certificate to add _certAddress
Output: true if certificate is added successfully, false otherwise

```
1  Begin
2      if (institution.checkInstitutePermission(msg.sender) != true) then// must be added by an authorized education institution
3          revert: "caller does not have permission"
4      end if
5      found ← false // initialize variable found to false
6      for i ← 0 to certificateCount − 1 do// iterates through the existing certificates to check that the added do not already exist
7          if (Strings.equal(certificates[i].ipfsHash, = _ipfsHash)) then // matches the added hash, _ipfsHash, with the existing
8              found ← true // set variable found to true
9              break // exit the loop
10         end if
11     end for
12     if (found = true) then// if cert already exist print error and stop execution
13         revert: "the certificate already exist"
14     end if
15     certificates[certificateCount] ← Certificate(ch, sa, ia, gd, sn, cd, r, et, y) //create new Certificate object and add to mapping
16     emit certificateGenerated(_ipfsHash) // emit event indicating a new certificate, with given hash, has been generated
17     return true // certificate is added successfully
18 End
```

exist, in line 15, an object of the *Certificate* struct is instantiated with the provided certificate details, and appended to the array of certificates in the *mapping* data structure. This results in a change to the ledger's global state, which is then communicated across the network to be approved by peer nodes. Finally, in line 16, an event is emitted to indicate that the certificate with the given hash is added to the blockchain. Through this event the hash is returned to the DApp, then to the educational institution, where it is shared with the graduate along with the actual digital certificate.

When there is a need to revoke a certificate, the *revokeCertificate()* function handles the revocation process. During revocation, the DApp receives the certificate information from the educational institution seeking to revoke the certificate through the interface shown in Fig. 12. This transaction must be signed with the institution's private key using their wallet.

The DApp extracts the hash from the provided certificate information and searches for it on the blockchain to process the revocation. It then invokes the *revokeCertificate()* function, passing the hash value as an argument. Algorithm 4 shows the function's execution process in revoking a certificate. As shown in lines 2-4, the function begins by verifying the revoking institution's permissions to ensure that only authorized institutions can revoke a certificate. This is accomplished by the *checkInstitutePermission()* function from the *Institution* contract, utilizing the educational institution's blockchain address as a parameter. In lines 5-11, it searches in the certificates *mapping* to find if the certificate to be revoked exists. If its hash and timestamp are found (in line 6), the certificate's revocation attribute is set to true in line 7, indicating that the certificate is revoked. Then, in line 8, it emits an event to log and send the revocation information to the DApp. This information is rendered to the educational institution and the graduate whose certificate is revoked. Additionally, the DApp modifies the IPFS content associated with the revoked certificate to represent its current status.

As stated earlier, the final consumers of certificates are third parties, such as recruiters, employers, and others, who must verify their candidates' credentials to prevent forgery. During the application process, they receive certificates from the candidates (graduates) along with their unique IDs or hashes. After that, the verifier enters the ID or hash into the DApp's verification interface in Fig. 13, and clicks the verification button to initiate the process. The DApp then invokes the *getCertDataById()* function, which is used for certificate verification, passing the certificate ID/hash as an argument. This function is marked with the keyword "*view*" to indicate that it does not need to be invoked through a transaction because it is a read-only function that does not change the ledger state. Since each node has a copy of the ledger state, this function executes on the local node, and it is open for anyone to use for certificate verification, eliminating the need for verifiers to be registered. Since it is executed

Algorithm 4 Revoking a certificate

Input: Hash of the certificate to be revoked _ipfsHash
Output: Certificate is revoked (through event emission)

```
1  Begin
2      if (institution.checkInstitutePermission(msg.sender) != true) then// check permission of the education institution
3          revert: "the caller is not authorized to perform this transaction"
4      end if
5      for i ← 0 to certificateCount − 1 do // iterates through existing certificates to check if the cert to be revoked exists
6          if(Strings.equal(certificates[i].ipfsHash, = _ipfsHash)) && (certificates[i].creation_date).length != 0)
           then//check if exist
7              certificates[i].revoked ← true //if found revoke by setting its revocation attribute to true
8              emit certificateRevoked(certificates[i].ipfsHash) // emit event indicating certificate is revoked
9              break // exit the loop
10         end if
11     end for
12     if(i = certificateCount) then// if not found
13         revert: "the certificate does not exist"
14     end if
15 End
```

locally and does not involve a transaction, the verifiers (e.g., employers) incur no cost for certificate verification.

Algorithm 5 represents the *getCertDataById()* function's execution process in verifying a certificate. In line 2, it initializes a temporary variable to store the certificate data, and in lines 3-4, it iterates through an array of certificates, checking for a match with the provided hash and ensuring its creation date exists. If a match is found, in line 5, it stores the certificate data on the temporary variable, and in line 6, it exits the loop; otherwise, in lines 9-11, it reverts with an error message indicating that the certificate is not verified. In line 12, the function then returns a tuple containing relevant certificate information, including the hash of the certificate, blockchain address of the graduate (certificate owner), blockchain address of the issuer, candidate's name, graduation date, award title (or description), and the revoked status. This information is returned to the DApp front-end for the verifier to view. Using the same hash, the DApp also retrieves the actual certificate document from the IPFS for the verifiers to preview. Note that the actual certificate stored in the IPFS contains the graduate's picture to help thwart impersonation.

The function *generateEqCertificate()* handles the process of issuing an equivalent certificate to the blockchain. Its workflow is similar to the *generateCertificate()* function. However, since equivalent certificates are issued by the regulatory authorities, this function can only be invoked by a regulatory authority. The same applies to the *revokeEqCertificate()* function; it works similarly to the *revokeCertificate()* function, but the only difference is that, it can only be called by a regulatory authority. Finally, the *getMyCertificates()* function uses the graduate's blockchain address to retrieve all certificates a graduate holds. It is used by graduates to access and view their certificate portfolio.

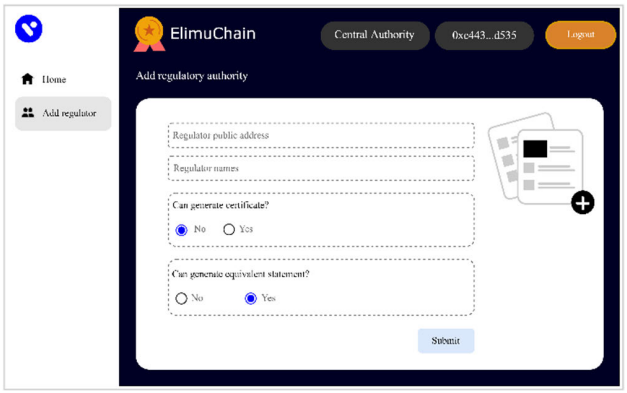

FIGURE 9. Interface for registering a regulatory authority.

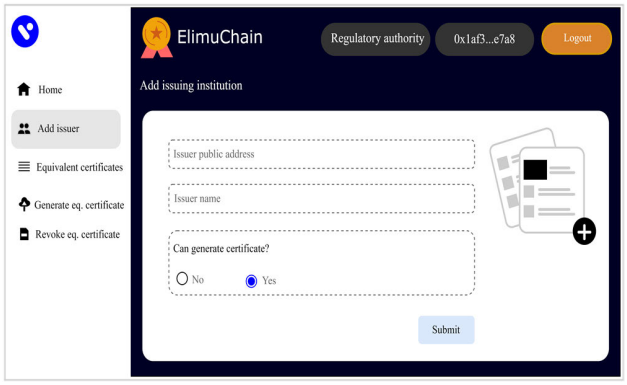

FIGURE 10. Interface for registering an issuing institution.

C. USER INTERFACES

Users interact with the system via various interfaces as shown in the screenshots, Fig. 9, Fig. 10, Fig. 11, Fig. 12, and Fig. 13.

Algorithm 5 Verifying a certificate

Input: Hash of the certificate to be revoked _hashId
Output: Certificate data (hash, owner address, issuer address, owner name, graduation date, revoked status)

```
Begin
    Certificate temp //declare temp variable
    for i ← 0 to certificateCount − 1 do
        if(Strings.equal(certificates[i].ipfsHash, = _hashId) && (certificates[i].creation_date).length != 0) then
            temp ← certificates[i]
            break //exit the loop
        end if
    end for
    if(i = certificateCount) then// if not found
        revert: ''the certificate is not verified''
    end if
    return (temp.hash, temp.owner_address, temp.issuer_address, temp.owner_name, temp.graduation_date,
    temp.revoked)
End
```

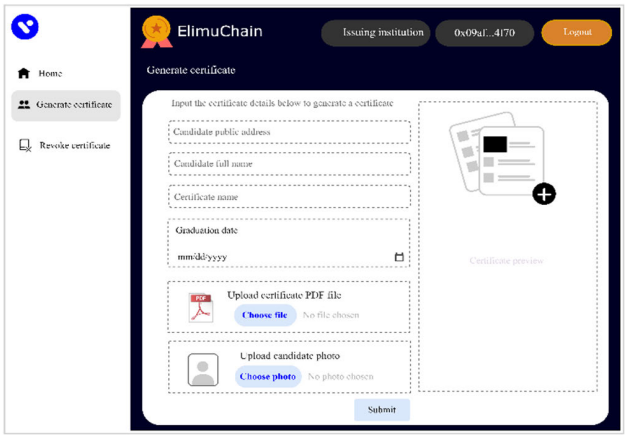

FIGURE 11. Interface for issuing a certificate.

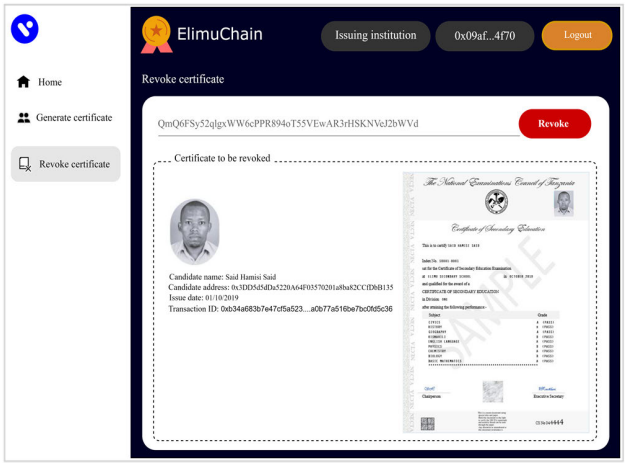

FIGURE 12. Interface for revoking a certificate.

D. DATA PROTECTION, PRIVACY, AND SECURITY

Our system design adheres to local and international data protection and privacy regulations, such as the General Data

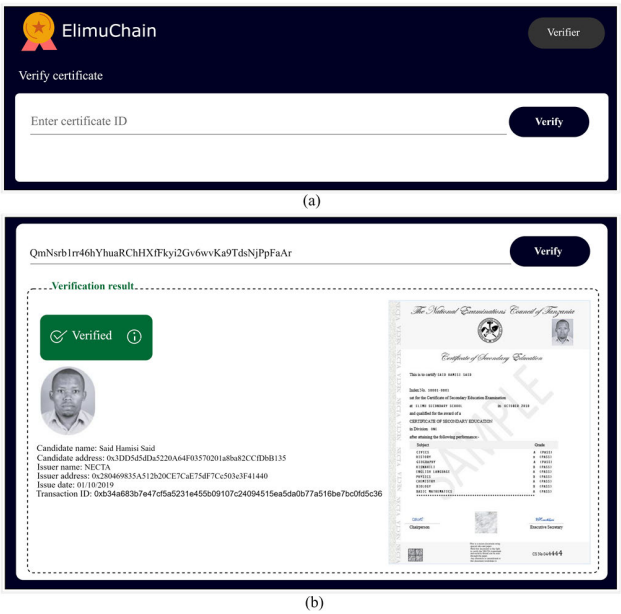

FIGURE 13. Interface for verifying a certificate: (a) Request to enter certificate ID, (b) Verification result.

Protection Regulation (GDPR) [117], California Consumer Privacy Act (CCPA) [118], and Tanzania's Personal Data Protection (Personal Data Collection and Processing) Regulations (PDPR) [119]. Though designed for specific regions, these regulations share a common goal of ensuring data protection and privacy. In compliance with these regulations, the system design accounts for the following. First, it incorporates the government regulatory authorities involved in the certification process in the country and lets them fulfil their responsibilities as mandated by the laws, such as, [64], [65], [66], [67], and [68]. This ensures regulatory oversight and enhances system security by deterring misuse. Second, it stores only hashes on the blockchain, not the actual

certificates. This approach preserves privacy and confidentiality by not exposing sensitive personal information. The immutable hashes in the ledger ensure certificate integrity, authenticity, and non-repudiation (preventing denial of issuance). Third, it gives graduates the autonomy to prove their certificate's authenticity without relying on the issuing institutions, giving them full control over their data. Certificate verification is possible only if they share their certificate IDs, allowing them to control with whom they share. Last, users in the system are identified by their blockchain addresses rather than real-world identities. This enhances anonymity and reduces personal data exposure and privacy breaches.

To ensure the identification, authorization, and authentication of the system users, we have implemented Role-Based Access Control and Permissions (RBAC-P) through smart contracts. It is implemented using *AcessControl* smart contract from Solidity libraries with the *_setupRole()* and *hasRole()* functions (see Fig. 5, Algorithms 1, 2, 3, and 4). This ensures that only authorized entities can join the blockchain and perform transactions like issuing certificates.

To ensure smart contract security, we followed best practices during implementation and conducted thorough code audits to identify and mitigate potential vulnerabilities. Adherence to security best practices included the following, as suggested in [59], [120], and [121]: (1) We used well-established standards, such as proven libraries like OpenZeppelin's *AccessControl*. These standards provide a framework for developing secure and reliable smart contracts. (2) We adhered to coding conventions, such as naming rules and explicit declarations of type, visibility, and access modifiers, to improve code quality, clarity, and consistency, which are critical for auditing, identifying, and mitigating security vulnerabilities. (3) We adopted a modular design approach, where each system's core functionality, such as registration, issuance, revocation, and verification, is implemented in separate modules, with distinct contracts and functions to minimize vulnerability risks.

We also performed a thorough review of the smart contracts using both manual auditing techniques and automated tools. In the manual audit, the code was carefully examined for potential weaknesses, logical inconsistencies, and deviations from best practices. On the other hand, tools such as Remix IDE, BscScan, and static analyzers were used to identify vulnerabilities, logical and syntactic errors, and potential risks.

This review focused on common blockchain and smart contract security vulnerabilities, such as the following:

- *Reentrancy attacks*: Occur when a malicious external contract repeatedly invokes a vulnerable contract before its initial execution is complete, potentially leading to logic manipulation, corruption of the contract state, and loopholes for malicious activities [59], [121]. To guard against these attacks, the ''checks-effects-interactions'' design pattern was applied to ensure that all state changes occur before any external call, interactions with external contracts were minimized to reduce the attack risks, and rigorous code testing was performed, as suggested in [59] and [121].
- *Integer overflows and underflows*: These vulnerabilities occur when arithmetic operations exceed the maximum or minimum limits of an integer type, exposing contracts to serious exploits [59], [121]. These risks, as suggested in [59], were mitigated by adopting secure coding practices and leveraging Solidity version 0.8.0's built-in safety features, which include overflow and underflow checks by default. Additional measures, included implementing conditions such as ''*require*'' statements to validate input values before performing arithmetic, utilizing proper data types and size (e.g., uint8, uint16, uint256, etc.), and verifying arithmetic operations.
- *Improper access controls*: Involve misconfiguration of functions, which may allow unauthorized parties to access or modify sensitive data [59], [121]. As suggested in [59], these issues were addressed by employing role-based access control and permission (RBAC-P) mechanisms using OpenZeppelin's *AccessControl* library. Audits were also performed to verify that sensitive functions and data are protected by role or permission checks and safeguarded against unauthorized calls.
- *Denial of service (DoS)*: Blockchain and smart contracts can face DoS attacks, where nodes are overwhelmed with high transaction volumes, leading to service disruptions [59], [121]. Fortunately, these attacks are mitigated through several inherent mechanisms: First, blockchain's decentralization and distribution nature allows duplication of workload across multiple nodes, ensuring no single point of failure during a DoS attack. Second, gas fees (a fee required for every transaction) deter attackers by making spammy function calls economically unviable. Additionally, block gas limits, which restrict the total gas usage per block, prevent functions from monopolizing resources through excessive computations, effectively deterring DoS attacks. Third, the BSC network, which uses PoSA, selects validators based on their stake and reputation, economically incentivizes them to uphold network integrity, and penalizes them for malicious behavior, creating economic barriers for attackers.
- *Majority attacks*: Occur when malicious actors gain control of more than half of the network's mining, staking, or voting power, enabling them to manipulate the network for their interests [59], [121]. However, in the BSC network, such attacks are mitigated through the PoSA consensus mechanism, which combines economic costs, reputational risks, and slashing penalties. Because the PoSA uses reputable (well-known) validators who stake a significant amount of cryptocurrency to participate in consensus, participating in such attacks is highly unlikely due to the prohibitive expenses. But also, they are afraid of losing their staked funds through slashing penalties if they participate in such attacks.

- *Sybil attacks*: Occur when attackers create multiple fake identities to gain undue influence over network consensus [37], [60]. The same mechanism, i.e., economic costs, reputational risks, and slashing penalties, which are used in *majority attacks* are also applicable in deterring *Sybil attacks.*
- *Uninitialized storage pointers:* Occur when uninitialized local storage variables inadvertently reference other storage variables in the contract, potentially creating vulnerabilities that attackers can exploit [59]. As suggested in [59], these vulnerabilities were addressed by ensuring that all storage variables are properly initialized before compilation. After all, the EVM compiler issues a warning for uninitialized variables.
- *Short address/parameter attacks*: Occur when a parameter sent to the smart contract is shorter than the expected parameter length, making the EVM append zeros to the end of the encoded parameter, potentially leading to vulnerabilities [59]. These attacks are prevented on the application side by ensuring that all arguments are properly validated before being sent to the blockchain, as suggested in [59].

As previously stated, our system leverages IPFS to address blockchain storage and computational limitations. Its decentralized and distributed nature reduces a single point of failure risk and enhances resilience against data loss and tampering [99], [100]. It is content-addressable, meaning files are identified by their cryptographic hashes (CID) [99], ensuring file integrity, tamper-proofing, and privacy. It also replicates data across multiple nodes, improving availability and fault tolerance. The IPFS divides data into chunks and stores them in separate nodes, strengthening security by improving data integrity, enhancing immutability, and reducing the risk of attacks like DoS [100]. Node unavailability is also a notable concern in IPFS. However, this problem has been evaded by using pining services, as suggested in [100].

VI. EXPERIMENTAL EVALUATION

A. EXPERIMENTAL DESIGN

Experiments were conducted to evaluate the DApp's performance and assess its suitability for national-level adoption in managing and verifying certificates to combat forgery. The DApp was evaluated on various aspects, including its economic viability, particularly the cost in cryptocurrency for performing transactions, and its performance efficiency in terms of throughput and latency. The experiments were carried out on the BSC Testnet, with BscScan used to provide statistics on transactions and other activities on the network. They were performed on a Dell-Latitude E6420 with an Intel Core i5-2520M CPU @ 2.50 GHz, 2 Core(s), 4 Logical Processor(s), 8 GB of RAM, and was running Windows 10, 62-bit OS. The evaluation results serve as a key reference for government authorities, policymakers, and other stakeholders in guiding the adoption of blockchain technology to combat fake certificates.

B. RESULTS AND DISCUSSION

1) THE COST OF USING ELIMUCHAIN DAPP

Transactions in blockchain systems, especially those involving smart contracts, typically incur costs, paid by users in the blockchain's native cryptocurrency, such as BNB for Binance Smart Chain. These costs usually include charges for executing the codes in the EVM, storing data in the blockchain, and network fees for processing and validating transactions. Therefore, assessing the feasibility of the cost is crucial for the DApp. The cost was calculated using formula (1).

$$Transaction\ cost = Gas\ used \times Gas\ price \quad (1)$$

where '*gas*' is a unit representing the computational effort needed for various operations on a blockchain network, such as processing transactions or executing smart contracts, and '*gas price*' represents the amount, in native cryptocurrency, payable for a unit of gas. We started by calculating the cost of running each DApp functionality, such as deploying contracts, registering institutions, and issuing, revoking, and verifying certificates. We then multiplied the unit cost by the number of entities associated with each functionality to get the total cost for each function. For example, for registering educational institutions, we multiplied the average cost of registering an individual institution by the number of institutions at that particular education level, and for issuing certificates, the average cost for a single certificate was multiplied by the number of graduates per year in each education level. That way, we assessed the economic viability of running the DApp at a national level to handle certificates from secondary to university level. Data used to support the estimations is based on recent education statistics in Tanzania [122], [123], [124], as shown in Table 3.

TABLE 3. Certificate issuing institutions and annual graduates.

Education level	Certificate issuing Institutions	# of issuing institutions	# of graduates per year
University	Universities	60	57,742
TET	TET Institutions	488	7,531
VET	VETA	1	20,833
A-Level Sec School	NECTA	1	106,883
O-Level Sec School			572,338

As shown in Table 4, deploying smart contracts incurred a higher cost, amounting to around 0.03133619 BNB for the *Institution* contract and 0.03501858 BNB for the *Certification* contract. However, this cost is not recurring, as smart contracts are deployed only once. Registering a regulatory authority costs around 0.00162279 BNB, totaling 0.00486837 BNB for the three regulatory authorities in Tanzania: TCU, NACTVET, and NECTA. The average cost for registering an educational institution is 0.00144293 BNB. Multiplying this amount by the number of issuing institutions in Table 3, i.e., $0.00144293 \times (60 + 488 + 1 + 1)$, results in a total of 0.7936115 BNB. The cost of registering regulatory authorities and educational institutions is also a one-time expense, as they only need to be registered once.

TABLE 4. The cost charged for running individual functionalities of the ElimuChain Dapp.

Entity/User	Functionality/Activity	Contract	Function invoked	Gas Used	Avrg Txn Cost (BNB)
Government Entity	Deploying smart contracts	Institution	constructor()	3133619	0.03133619
		Certification	constructor()	3501858	0.03501858
	Registering regulatory authorities	Institution	addRegulator()	195481	0.00162279
Regulatory Authority	Registering Issuing institutions		addInstitute()	265265	0.00144293
	Issue equivalent certificates	Certification	registerEqCertificate()	453432	0.00226716
Educational Institution	Issuing certificates		registerCertificate()	418888	0.00209444
	Revoking certificates		revokeCertificate()	260705	0.00130352
Third-party (e.g., Employer)	Verifying certificates		getCertDataByID()	0	0.00000000
Graduate	Viewing certificates		getMyCertificates()	0	0.00000000

Certificates are issued annually, in every graduation year, making this a recurring cost that significantly impacts the economic viability of the DApp. As depicted in Fig. 15, it varies depending on the number of graduates in each education level. As described earlier, certificates are issued by invoking *registerCertificate()* function whose execution steps are shown in Algorithm 3. The average cost for issuing a certificate is 0.00209444 BNB, as indicated in Table 4. Multiplying this amount by the number of graduates at each education level (shown in Table 3), gives the total annual cost for issuing certificates for each education level, as depicted in Fig. 15. Ordinary-level Secondary School incurs the highest annual cost of 1198.73 BNB due to its large number of graduates, about 572,338 per year. In contrast, TET incurs a lower annual cost of 15.8 BNB because of its smaller number of graduates, around 7,531 per year.

An equivalent certificate, which is issued by invoking *registerEqCertificate()* function, incurs an average cost of 0.00226716 BNB (see Table 4). Its cost is slightly higher than that of regular certificates (see Fig. 14), but its overall impact is minimal because equivalent certificates are fewer than regular certificates, and their issuance is occasional.

Certificate revocation, which involves invoking the *revokeCertificate()* function (whose process is shown in Algorithm 4), incurs the least gas and cost, as shown in Table 4 and Fig. 14. Revocation also occurs rarely, so its total cost is insignificant.

As shown in Table 4 and Fig. 14, verifying certificates on ElimuChain is free, meaning that third parties (e.g., employers) will incur no cost for verifying certificates. Certificate verification is performed by calling the *getCertDataByID()* function, as outlined in Algorithm 5. This function is declared with the "*view" state mutability specifier*, making it read-only, incapable of performing transactions (write operations), and changing the blockchain state. For that reason, it does not require gas for its execution. This is possible because it is invoked from the local node, which maintains a copy of the ledger and can serve any non-state-changing query. The same applies to viewing certificates. As shown in Table 4 and Fig. 14 graduates are not charged any fee to view their certificates since it involves invoking a read-only, non-state-changing, and non-transactional function *getMyCertificates()* from a local node.

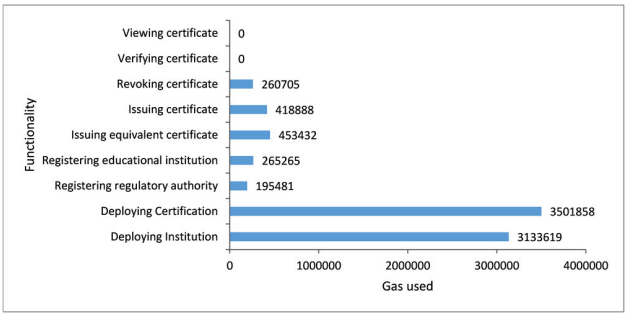

FIGURE 14. Gas used for each functionality.

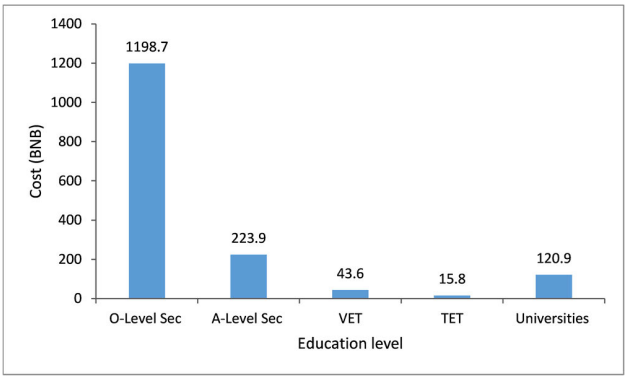

FIGURE 15. The cost of issuing certificates for each educational level.

Since our DApp stores actual certificate files on IPFS, we evaluated whether file size influences transaction costs, particularly for the certificate issuance and revocation functions. This experiment was conducted by repeating each transaction 30 times on 10 different certificate file sizes in Kilobytes (KB). In each iteration, the cost was recorded, and the average cost was calculated to determine the cost for each specific file size. This helps minimize variations, as the cost fluctuates due to changing network conditions. As shown in Fig. 16, a scatter plot and regression line that best fits the data were drawn for both issuing and revoking to show the relationship between file size and gas used. The figure clearly shows that issuance consumes more gas than revocation. This is because the *revokeCertificate()* function only updates the revocation status from false to true, requiring lesser computational efforts compared to *generateCertificate()* function, which needs to store certificate hash and other metadata on

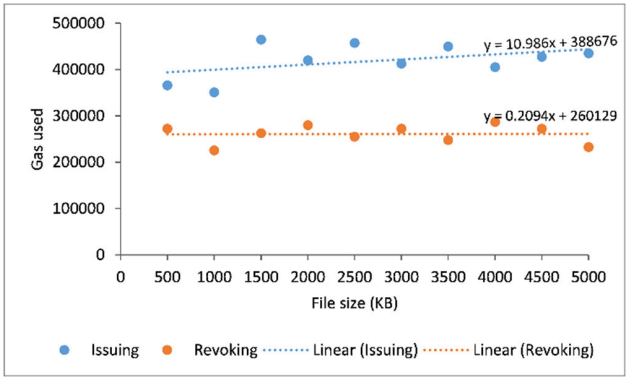

FIGURE 16. Gas usage per different file sizes in KB.

the ledger, which demands more computation. Regarding the influence of file size on gas usage, the regression lines represent the average rate of change in gas consumption as file size increases. Issuing is represented by the equation $y = 10.986x + 388676$, with a 10.986 slope. This means that for each increase in file size, the total gas for issuing a certificate rises by approximately 10.986 units. For revocation, the equation is $y = 0.2094x + 260129$, with a 0.2094 line slope. This means that as file size increases, the total gas usage for revoking a certificate increases by approximately 0.2094 units. These findings indicate that certificate file size has little impact on gas usage in transactions. This influence is mainly due to the increased computational, storage, and network resources required to process and store larger amounts of data on the blockchain. However, since certificate files are usually small, mostly in KBs, this influence becomes negligible with minimal practical impact.

From the data in Fig. 16, we also computed descriptive statistics i.e., mean, maximum, minimum, and standard deviation values, as shown in Table 5. The data show variability in gas usage for different file sizes, as indicated by the standard deviations. The maximum gas usage occurred at 1500 KB for issuance and 4000 KB for revocation, while the minimum was reached at 1000 KB for both.

TABLE 5. Descriptive data for the cost of issuing and revoking certificate.

Statistical measure	Issuing		Revoking	
	Gas used	Cost (BNB)	Gas used	Cost (BNB
Mean	418887.8	0.00209444	260705.0	0.00130352
Maximum	464671.0	0.00232335	287171.0	0.00143585
Minimum	350934.0	0.00175467	225326.0	0.00112663
Standard deviation	37302.21	0.00018651	20293.97	0.00010147

2) PERFORMANCE EFFICIENCY

One of the major barriers to blockchain adoption in various applications is its performance and scalability issues. Therefore, thorough performance testing is essential before adopting a DApp for real-world use. We evaluated the DApp's efficiency by measuring throughput and latency across its functionalities, i.e., registering regulatory authorities and educational institutions, as well as issuing, revoking, verifying, and viewing certificates.

Latency measures the time taken from initiating an operation to its completion. This includes transactional operations, such as writing to the ledger, and non-transactional operations, such as reading from the ledger. For transaction latency, BSC Fast Finality with BEP-126 was considered, which takes around 3 blocks confirmations with 3 seconds block time, leading to around 9 seconds for transaction finality. Finality time and confirmation time constitute total transaction latency. The experiment was conducted by repeating each operation 30 times to account for variations caused by network conditions. In each iteration, the time in seconds was recorded, and the average time was computed to obtain latency for a particular functionality. At the start of each operation, the initial time T_i was recorded, and upon completion, the completion time T_c was recorded, then latency was calculated as in (2).

$$\Delta T = T_c - T_i. \tag{2}$$

Fig. 17 shows the average latency for performing each functionality in the DApp. Viewing and verifying certificates have much lower latency compared to the rest of the functionalities. This is because, unlike others, these functionalities are non-transactional. They only retrieve and confirm data that is already stored on the blockchain, and they are usually served by a local node, which contains a copy of a ledger. This process is faster than writing data to the blockchain, which involves more steps. Overall, ElimuChain performs each functionality much faster than traditional and manual methods. For instance, certificate verification takes about 3.83 seconds, whereas traditional and manual methods can take weeks or months.

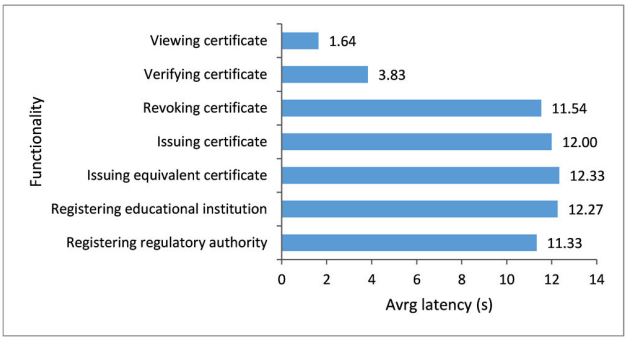

FIGURE 17. Average latency for each functionality in ElimuChain DApp.

We also examined whether file size influences the latency, particularly for certificate issuing, revoking, verifying, and viewing. This experiment was done on 10 different certificate file sizes in KB. Fig. 18 shows the experimental results, illustrating the relationship between file size and latency for the mentioned operations. As expected, issuance and revocation show higher latency due to their transactional nature, which involves multiple processing steps that can

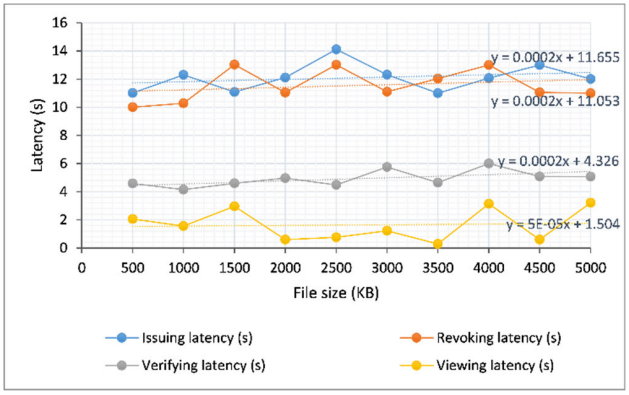

FIGURE 18. The influence of file size on latency for issuing, revoking, verifying, and viewing certificates.

be affected by network delays, queues, and fees. Contrary, verifying and viewing certificates exhibit low latency since they are non-transactional operations. Regarding file size influence, a regression line was fitted to the scatter plot, and its gradient was calculated to illustrate how latency changes with increasing file size. It turns out that issuance, revocation, and verification operations have a 0.0002 slope, meaning that as file size increases, their latency increases by about 0.0002. On the other hand, the viewing operation has a 5E-05 slope, indicating that as file size increases, the viewing latency increases by around 5E-05. In conclusion, as file size increases a little increase in latency was observed. However, the increase is insignificant with negligible impact in practical terms, considering that certificate files are typically small in size.

The variability in latency, as indicated by the mean, maximum, minimum, and standard deviation in Table 6, is mainly due to variations in network conditions, such as congestion and priority fees for miners. The maximum latency was observed at 2500KB for issuance, 1500KB for revocation, 4000KB for verification, and 5000KB for viewing. And they reached the minimum latency at 3500KB for issuance, 500KB for revocation, 1000KB for verification, and 3500KB for viewing.

TABLE 6. Descriptive stats for the latency of issuing, revoking, verifying, and viewing a certificate.

Statistical measure	Issuing latency (s)	Revoking latency (s)	Verifying latency (s)	Viewing latency (s)
Mean	12.11	11.56	3.94	1.639
Maximum	14.12	13.03	6.00	3.21
Minimum	11.00	10.01	4.15	0.29
Standard deviation	0.97	1.14	0.58	1.14

Throughput is the rate at which transactions or operations are processed in a given time, typically measured in transactions per second (TPS) or operations per second (OPS). It measures the system's capacity to handle many requests and perform operations efficiently. The throughput was computed for the issuing, revoking, verifying, and viewing certificate operations by repeating each operation 30 times for 9 different loads, 2 to 512 certificates. In each transaction or operation, the execution time was calculated using formula (2), its average was computed, and the throughput for each batch was calculated by dividing the number of transactions or operations by the average time taken to complete the execution, as shown in formula (3).

$$Throughput\ (TPS) = \frac{Number\ of\ transactions}{Time\ taken\ (seconds)} \tag{3}$$

As shown in Fig. 19, the throughput increases as the number of certificates increases, demonstrating that the system is efficient and scalable, i.e., scales well with the increasing load. The result also indicates that non-transactional operations, verifying certificates and viewing certificates, have higher throughput, efficiency, and scalability, than transactional operations, issuing and revoking certificates. This is because the former do not involve mining process, whereas the later do.

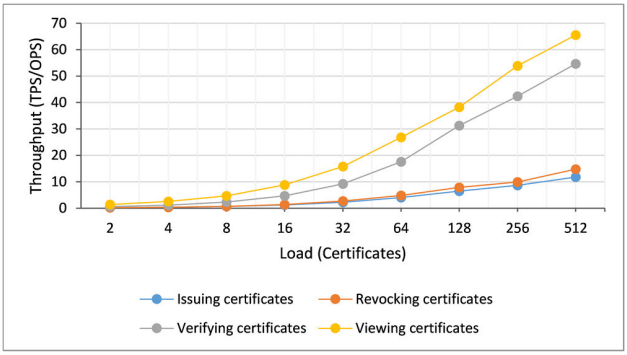

FIGURE 19. Throughput of issuing, revoking, verifying, and viewing certificates.

The performance results of our system were compared with the previous work of Serranito [61] in terms of the average throughput and latency for the basic operations, i.e., issuing, revoking, and verifying certificates. Although Serranoto's work uses Ethereum, which has a different underlying architecture from the BSC used in our study, it was chosen because, unlike other works, it provides thorough evaluation results with similar functionalities and metrics that can be benchmarked with our work. The functionalities and metrics for comparison were chosen based on their similarities to our work.

As shown in Fig. 20 and Fig. 21, our system demonstrates lower latency and higher throughput for the transactional operations of issuing and revoking certificates. This is because our solution uses BSC with PoSA, which is more efficient and scalable than its counterpart. For the non-transactional operation of verifying certificates, our solution has slightly higher latency and lower throughput. This can be attributed to our solution covering all education levels, whereas Serranito's work focuses only on universities. Additionally, our solution provides a full preview of the actual certificates during verification. Finally, non-transactional operations like certificate verification depend

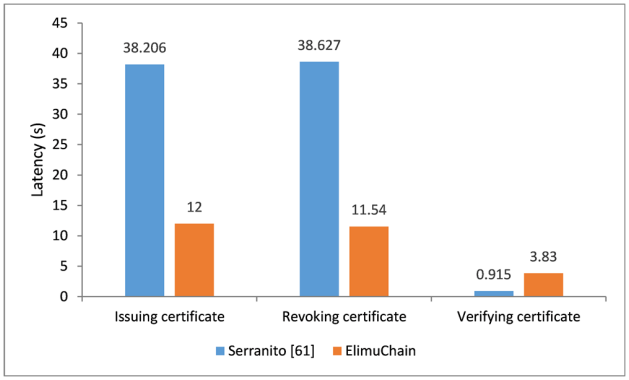

FIGURE 20. Latency comparison with prior work.

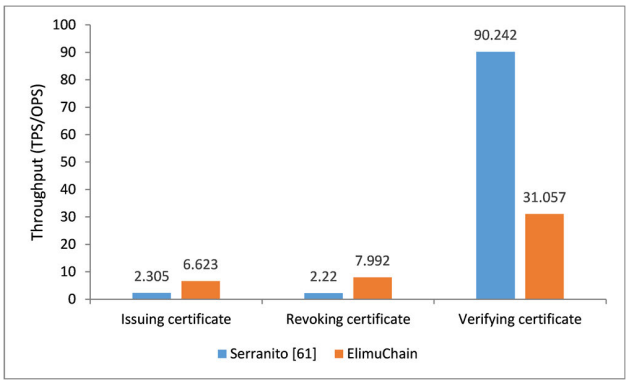

FIGURE 21. Throughput comparison with prior work.

largely on the capabilities of the local node used in the operation.

VII. CONCLUSION, LIMITATIONS, AND FUTURE WORK

A. CONCLUSION

In response to widespread educational certificate forgery and the lack of holistic and effective solutions, this study proposes a comprehensive blockchain-enabled system for issuing, revoking, and verifying certificates. The system is tailored to the Tanzanian education ecosystem, where its applicability was tested. As a proof of concept, the system is actualized by developing a DApp, ElimuChain, using BSC blockchain, smart contracts, and IPFS. At its core, it generates a certificate hash linked to the graduate's and issuer's IDs, stores it on the blockchain, and stores the actual certificate on IPFS, enabling verifiers to query the hash from the blockchain and the actual certificate from the IPFS for verification. Our solution is holistic in the sense that it can be rolled out nationwide to address certificates from all education levels and institutions in the country, providing a one-stop center for verifying all certificates. Despite the blockchain's inherent limitations in performance, scalability, storage, and computation, this study has achieved its large-scale implementation by forging a synergy between blockchain, smart contracts, and IPFS. This integration enables off-chain storage and computation of large data, reducing the burden on the blockchain and enhancing overall system performance. It also takes advantage of the BSC PoSA consensus protocol, which combines DPoS and PoA to achieve high performance, throughput, scalability, and cost-effectiveness. Apart from issuing and verifying certificates, the DApp includes authority management and Role Based Access Control and Permission (RBAC-P) to monitor the entities involved in the system, thereby preventing system misuse. It features an on-chain certificate revocation mechanism through smart contracts, which is more efficient than existing solutions, and leverages IPFS to handle large data, overcoming blockchain storage and computational limits. Furthermore, it supports the verification of foreign awards by accommodating equivalency certificates. The properties of blockchain, including decentralization, distribution, immutability, tamper-proof, anonymity, disintermediation, and smart contracts, have been critical in enabling its functionality.

The system is deployed on the BSC blockchain to evaluate it for latency, throughput, scalability, and operational costs, especially for nationwide deployment to handle certificates from all educational levels and institutions. The results show that the average cost for issuing and revoking a certificate is trivial, approximately 0.00209444 and 0.00130352 BNB, respectively, while verifying a certificate is free, meaning that verifiers incur no cost for verification. For latency, it performs issuance, revocation, and verification in an average of 12.00, 11.54, and 3.83 seconds, respectively. Its throughput for issuing, revoking, and verifying certificates is 6.623, 7.992, and 31.057 TPS, respectively. These results, particularly throughput and latency, were compared with the previous solution, and our system outperforms the counterpart for transactional operations. Overall, the solution proves to be cost-effective, scalable, and efficient for managing and verifying certificates.

B. LIMITATIONS AND FUTURE WORK

While this study offers valuable contributions, it also has some limitations that warrant acknowledgment and consideration in future research.

First, this study focuses on educational certificates because they play a crucial role in providing social and economic opportunities for individuals, making them frequent targets of forgery. As mentioned earlier, forgery of these documents has a widespread impact including, undermining the credibility of the education system, disrupting job markets, and compromising workforce quality, the economy, and public welfare. However, the proposed solution may also apply to other documents, such as licenses, identity, or legal certificates. Future research could explore the applicability of this solution to other document types.

Second, this study is limited to developing and testing ElimuChain as a prototype system in a simulated environment using the testnet with artificial cryptocurrencies and certificates to demonstrate its feasibility. It does not include full-implementation and integration with legacy systems in real schools or institutions. Future research should explore

deploying the system on the mainnet with actual cryptocurrencies, integrating it with legacy systems, and evaluating its broader impact in real-world contexts through pilot testing in collaboration with government and educational institutions.

Third, the technical implementation and experimental evaluation of the proposed system are utterly limited to the BSC blockchain platform, IPFS, and smart contracts environment. However, with the emergence of various other blockchain technologies, future work should focus on deploying the system on different blockchain platforms to assess its performance and adaptability in different environments.

Fourth, in this study, the cost evaluation is limited to the DApp's operational expenses, including the cost of transactions, storage, computations, and smart contract execution operations, all of which are necessary for processing and validating transactions on the blockchain network. Future research should expand the cost evaluation to make it more comprehensive and include other costs, such as infrastructure, deployment and implementation, maintenance and support, and security and compliance costs.

Last, this study did not compare the cost of this solution with the cost involved in the traditional paper-based certificates. These costs are expected to be considerably higher and difficult to determine because they involve labor or payroll costs, paperwork, printing machinery, administrative costs, transport costs, and other logistic costs. A study should be conducted to thoroughly investigate these costs and compare them with the costs involved in the proposed solution.

REFERENCES

[1] M. Baldi, F. Chiaraluce, M. Kodra, and L. Spalazzi, "Security analysis of a blockchain-based protocol for the certification of academic credentials," 2019, *arXiv:1910.04622*.

[2] C.-S. Hsu, S.-F. Tu, and P.-C. Chiu, "Design of an e-diploma system based on consortium blockchain and facial recognition," *Educ. Inf. Technol.*, vol. 27, no. 4, pp. 5495–5519, Jan. 2022, doi: 10.1007/s10639-021-10840-5.

[3] N. Smolenski. (2016). *Academic Credentials in an Era of Digital Decentralization*. Learn. Mach. Res., Anaheim, CA, USA. [Online]. Available: https://www.academia.edu/29403234/ademic_Credentials_in_an_Era_of_Digital_Decentralization

[4] C. Johnson, "Credentialism and the proliferation of fake degrees: The employer pretends to need a degree; the employee pretends to have one," *Hofstra Labor Employ. Law J.*, vol. 23, no. 2, pp. 1–75, Aug. 2006.

[5] F. R. Vidal, F. R. Gouveia, and C. Soares, "Blockchain application in higher education diploma management and results analysis," *Adv. Sci. Technol. Eng. Syst.*, vol. 5, no. 6, pp. 871–882, Dec. 2020, doi: 10.25046/aj0506104.

[6] M. E. Effiong, "A framework for the adoption of blockchain technology in academic certificate-verification systems: A case study in Nigeria," M.S. thesis, Dept. School Information Technology, Tallinn Univ. Technol., Tallinn, Estonia, 2020.

[7] S. O. Ghazali, O. Rana, and M. Ehsan, "Blockchain based framework for educational certificates verification," *J. Crit. Rev.*, vol. 7, no. 3, pp. 79–84, Mar. 2020, doi: 10.31838/jcr.07.03.13.

[8] A. Ezell and J. Bear, *Degree Mills: The Billion-Dollar Industry That Has Sold Over a Million Fake Diplomas*. New York, NY, USA: Prometheus, 2012.

[9] V. Wahi, A. K. Cherukuri, K. Srinivasan, and A. Jonnalagadda, "CryptoCert: A blockchain-based academic credential system," in *Recent Trends in Blockchain for Information Systems Security and Privacy*, 1st ed., A. K. Tyagi and A. Abraham, Eds., London, U.K.: Taylor & Francis, 2021, p. 21, doi: 10.1201/9781003139737-19.

[10] P. Sengati and E. Kitinya, "An inquiry on strategies to mitigate fake certificates: A case of Tanzania," *Int. J. Develop. Res.*, vol. 8, no. 12, pp. 24875–24881, Dec. 2018. [Online]. Available: https://www.journalijdcom/inquiry-strategies-mitigate-fake-certificates-case-tanzania

[11] Citizen. (2017). *Your Bus. is Our Business: Fake Certificates in TZ Economic Equation*. Accessed: Jun. 9, 2022. [Online]. Available: https://w.thecitizen.co.tz/tanzania/magazines/your-business-is-our-busins-fake-certificates-in-tz-economic-equation-2590982

[12] P. Oblikwu and K. Dekera, "A generic certificate verification system for Nigerian universities," *Int. J. Comput. Sci. Mobile Comput.*, vol. 8, no. 10, pp. 137–148, Oct. 2019. [Online]. Available: https://ijcsmc.m/docs/papers/October2019/V8I10201924.pdf

[13] G. Grolleau, T. Lakhal, and N. Mzoughi, "An introduction to the economics of fake degrees," *J. Econ. Issues*, vol. 42, no. 3, pp. 673–693, Sep. 2008. [Online]. Available: https://halshs.chives-ouvertes.fr/halshs-00326238.

[14] E. B. Cohen and R. Winch, "Diploma and accreditation mills: New trends in credential abuse," Verifile Ltd. Accredibase, Bedford, U.K., 1853 Tech. Rep., 2011. Accessed: Dec. 3, 2023. [Online]. Available: https://etico.iiep.unesco.org/en/diploma-and-accreditation-mills-new-trends-credential-abuse

[15] I. Gowhar. (2017). *Degree Certificate Racket Thrives in Bengaluru*. Hindu. Accessed: Jun. 9, 2022. [Online]. Available: https://www.thehindu.com/news/ties/bangalore/degree-certificate-racket-thrives-in-bengaluru/article127959.ece?utm_content=buffer69bc3&utm_medium=social&utm_source=twitter.com&utm_campaign=buffer

[16] H. Clifton, M. Chapman, and S. Cox. (2018). *'Staggering' Trade in Fake Degrees Revealed*. BBC News. Accessed: Jul. 13, 2023. [Online]. Available: https://www.bbc.com/news/uk-42579634

[17] New York Times. (2015). *A Rising Tide of Bogus Degrees*. Accessed: Jul. 13, 2023. [Online]. Available: https://www.nytimes.com/15/05/20/opinion/a-rising-tide-of-bogus-degrees.html

[18] A. Tariq, H. B. Haq, and S. T. Ali, "Cerberus: A blockchain-based accreditation and degree verification system," *IEEE Trans. Computat. Social Syst.*, vol. 10, no. 4, pp. 1503–1514, Aug. 2023, doi: 10.1109/TCSS.2022.3188453.

[19] E. C. Garwe, "Qualification, award and recognition fraud in higher education in Zimbabwe," *J. Stud. Educ.*, vol. 5, no. 2, pp. 119–135, Apr. 2015, doi: 10.5296/jse.v5i2.7456.

[20] D. Maji, R. S. Lamkoti, H. Shetty, and B. Gondhalekar, "Certificate verification using blockchain and generation of transcript," *Int. J. Eng. Res. Technol.*, vol. 10, no. 3, pp. 539–544, Apr. 2021. [Online]. Available: https://www.ijert.org/research/certificate-verification-usinblockchain-and-generation-of-transcript-IJERTV10IS030260.pdf

[21] F. Schär and F. Mosli, "Blockchain diplomas using smart contracts to secure academic credentials," *Beiträge zur Hochschulforsch*, vol. 41, pp. 48–58, Oct. 2019. [Online]. Available: https://www.researgate.net/publication/334098551

[22] S. H. Said, M. A. Dida, E. M. Kosia, and R. S. Sinde, "A blockchain-based conceptual model to address educational certificate verification challenges in Tanzania," *Eng., Technol. Appl. Sci. Res.*, vol. 13, no. 5, pp. 11691–11704, Oct. 2023, doi: 10.48084/etasr.6170.

[23] D.-H. Nguyen, D.-N. Nguyen-Duc, N. Huynh-Tuong, and H.-A. Pham, "CVSS: A blockchainized certificate verifying support system," in *Proc. 9th Int. Symp. Inf. Commun. Technol.-SoICT*, Da Nang City, Vietnam, Dec. 2018, pp. 436–442, doi: 10.1145/3287921.3287968.

[24] R. Xie, Y. Wang, M. Tan, W. Zhu, Z. Yang, J. Wu, and G. Jeon, "Ethereum-blockchain-based technology of decentralized smart contract certificate system," *IEEE Internet Things Mag.*, vol. 3, no. 2, pp. 44–50, Jun. 2020, doi: 10.1109/IOTM.0001.1900094.

[25] D. Oliveira, M. Rahouti, A. Jaesim, N. Siasi, and L. Ko, "Can the inter planetary file system become an alternative to centralized architectures?" in *Proc. Hum. Interact., Emerg. Technol. Future Syst.*, vol. 319, T. Ahram and R. Taiar, Eds., Cham, Switzerland: Springer, Sep. 2021, pp. 597–604, doi: 10.1007/978-3-030-85540-6_75.

[26] J. Gresch, B. Rodrigues, E. J. Scheid, S. S. Kanhere, and B. Stiller, "The proposal of a blockchain-based architecture for transparent certificate handling," in *Proc. 21st Int. Conf. Bus. Inf. Syst.*, vol. 339, Berlin, Germany, Jan. 2019, pp. 185–196, doi: 10.1007/978-3-030-04849-5_16.

[27] N. Nizamuddin, K. Salah, M. Ajmal Azad, J. Arshad, and M. H. Rehman, "Decentralized document version control using Ethereum blockchain and IPFS," *Comput. Electr. Eng.*, vol. 76, pp. 183–197, Jun. 2019, doi: 10.1016/j.compeleceng.2019.03.014.

[28] N. Mahmoud, A. Aly, and H. Abdelkader, "Enhancing blockchain-based ride-sharing services using IPFS," *Intell. Syst. Appl.*, vol. 16, Nov. 2022, Art. no. 200135, doi: 10.1016/j.iswa.2022.200135.

[29] G. Caldarelli and J. Ellul, "Trusted academic transcripts on the blockchain: A systematic literature review," *Appl. Sci.*, vol. 11, no. 4, p. 1842, Feb. 2021, doi: 10.3390/app11041842.

[30] R. H. Sayed, "Potential of blockchain technology to solve fake diploma problem," M.S. thesis, Dept. Computer Science Information Systems, Jyväskylä Univ., Jyväskylä, Finland, 2019.

[31] A. Sedik, Y. Maleh, G. M. E. Banby, A. A. M. Khalaf, F. E. A. El-Samie, B. B. Gupta, K. Psannis, and A. A. A. El-Latif, "AI-enabled digital forgery analysis and crucial interactions monitoring in smart communities," *Technological Forecasting Social Change*, vol. 177, Apr. 2022, Art. no. 121555, doi: 10.1016/j.techfore.2022.121555.

[32] M. Emam, Q. Han, L. Yu, and H. Zhang, "A keypoint-based region duplication forgery detection algorithm," *IEICE Trans. Inf. Syst.*, vol. 99, no. 9, pp. 2413–2416, Jan. 2016, doi: 10.1587/transinf.2016edl8024.

[33] M. Emam, Q. Han, Q. Li, H. Zhang, and M. Emam, "A robust detection algorithm for image copy-move forgery in smooth regions," in *Proc. Int. Conf. Circuits, Syst. Simulation (ICCSS)*, Jul. 2017, pp. 119–123, doi: 10.1109/CIRSYSSIM.2017.8023194.

[34] M. Emam, Q. Han, and X. Niu, "PCET based copy-move forgery detection in images under geometric transforms," *Multimedia Tools Appl.*, vol. 75, no. 18, pp. 11513–11527, Sep. 2015, doi: 10.1007/s11042-015-2872-2.

[35] M. Emam, Q. Han, and H. Zhang, "Two-stage keypoint detection scheme for region duplication forgery detection in digital images," *J. Forensic Sci.*, vol. 63, no. 1, pp. 102–111, Jan. 2018, doi: 10.1111/1556-4029.13456.

[36] M. Zanardelli, F. Guerrini, R. Leonardi, and N. Adami, "Image forgery detection: A survey of recent deep-learning approaches," *Multimedia Tools Appl.*, vol. 82, no. 12, pp. 17521–17566, May 2023, doi: 10.1007/s11042-022-13797-w.

[37] J. Andrew, D. P. Isravel, K. M. Sagayam, B. Bhushan, Y. Sei, and J. Eunice, "Blockchain for healthcare systems: Architecture, security challenges, trends and future directions," *J. Netw. Comput. Appl.*, vol. 215, Jun. 2023, Art. no. 103633, doi: 10.1016/j.jnca.2023.103633.

[38] K. Rajeshkumar, C. Ananth, and N. Mohananthini, "Blockchain-assisted homomorphic encryption approach for skin lesion diagnosis using optimal deep learning model," *Eng. Technol. Appl. Sci. Res.*, vol. 13, no. 3, pp. 10978–10983, Jun. 2023, doi: 10.48084/etasr.5594.

[39] I. Ahmad, S. Abdullah, and A. Ahmed, "IoT-fog-based Healthcare 4.0 system using blockchain technology," *J. Supercomput.*, vol. 79, no. 4, pp. 3999–4020, Mar. 2023, doi: 10.1007/s11227-022-04788-7.

[40] Z. Chen, Z. Zhu, Z.-J. Wang, and Y. P. Tsang, "Fairness-aware large-scale collective opinion generation paradigm: A case study of evaluating blockchain adoption barriers in medical supply chain," *Inf. Sci. (Ny).*, vol. 635, pp. 257–278, Mar. 2023, doi: 10.1016/j.ins.2023.03.135.

[41] N. Yadav, S. Luthra, and D. Garg, "Blockchain technology for sustainable supply chains: A network cluster analysis and future research propositions," *Environ. Sci. Pollut. Res.*, vol. 30, no. 24, pp. 64779–64799, Apr. 2023, doi: 10.1007/s11356-023-27049-3.

[42] Z.-J. Wang, Z.-S. Chen, L. Xiao, Q. Su, K. Govindan, and M. J. Skibniewski, "Blockchain adoption in sustainable supply chains for industry 5.0: A multistakeholder perspective," *J. Innov. Knowl.*, vol. 8, no. 4, Oct. 2023, Art. no. 100425, doi: 10.1016/j.jik.2023.100425.

[43] Z.-J. Wang, Y. Sun, Q. Su, M. Deveci, K. Govindan, M. J. Skibniewski, and Z.-S. Chen, "Smart contract application in resisting extreme weather risks for the prefabricated construction supply chain: Prototype exploration and assessment," *Group Decis. Negotiation*, vol. 33, no. 5, pp. 1049–1087, Oct. 2024, doi: 10.1007/s10726-024-09877-x.

[44] A. Ahmed, S. Abdullah, S. Iftikhar, I. Ahmad, S. Ajmal, and Q. Hussain, "A novel blockchain based secured and QoS aware IoT vehicular network in edge cloud computing," *IEEE Access*, vol. 10, pp. 77707–77722, 2022, doi: 10.1109/ACCESS.2022.3192111.

[45] D. Shao, C. Kombe, and S. Saxena, "An ensemble design of a cash crops-warehouse receipt system (WRS) based on blockchain smart contracts," *J. Agribusiness Developing Emerg. Economies*, vol. 13, no. 5, pp. 762–774, Nov. 2023, doi: 10.1108/JADEE-02-2022-0032.

[46] N. Elisa, L. Yang, F. Chao, N. Naik, and T. Boongoen, "A secure and privacy-preserving E-government framework using blockchain and artificial immunity," *IEEE Access*, vol. 11, pp. 8773–8789, 2023, doi: 10.1109/ACCESS.2023.3239814.

[47] H. A. M. Deenmahomed, M. M. Didier, and R. K. Sungkur, "The future of university education: Examination, transcript, and certificate system using blockchain," *Comput. Appl. Eng. Educ.*, vol. 29, no. 5, pp. 1234–1256, Sep. 2021, doi: 10.1002/cae.22381.

[48] A. Grech and A. F. Camilleri, "Blockchain in education," Eur. Commission JRC, Seville, Spain, Tech. Rep. EUR 28778 EN, 2017, doi: 10.2760/60649.

[49] H. Taherdoost, "Blockchain technology and artificial intelligence together: A critical review on applications," *Appl. Sci.*, vol. 12, no. 24, p. 12948, Dec. 2022, doi: 10.3390/app122412948.

[50] N. K. Al-Shammari, T. H. Syed, and M. B. Syed, "An edge—IoT framework and prototype based on blockchain for smart healthcare applications," *Eng., Technol. Appl. Sci. Res.*, vol. 11, no. 4, pp. 7326–7331, Aug. 2021, doi: 10.48084/etasr.4245.

[51] A. Ahmed, S. Abdullah, M. Bukhsh, I. Ahmad, and Z. Mushtaq, "An energy-efficient data aggregation mechanism for IoT secured by blockchain," *IEEE Access*, vol. 10, pp. 11404–11419, 2022, doi: 10.1109/ACCESS.2022.3146295.

[52] M. R. Dorsala, V. N. Sastry, and S. Chapram, "Blockchain-based solutions for cloud computing: A survey," *J. Netw. Comput. Appl.*, vol. 196, Dec. 2021, Art. no. 103246, doi: 10.1016/j.jnca.2021.103246.

[53] H. A. Ahmed and J. Jang, "Higher educational certificate authentication system using QR code tag," *Int. J. Appl. Eng. Res.*, vol. 12, no. 20, pp. 9728–9734, 2017. [Online]. Available: https://www.ripublicati.com/ijaer17/ijaerv12n20_64.pdf

[54] R. Q. Castro and M. Au-Yong-Oliveira, "Blockchain and higher education diplomas," *Eur. J. Invest. Health, Psychol. Educ.*, vol. 11, no. 1, pp. 154–167, Feb. 2021, doi: 10.3390/ejihpe11010013.

[55] A. W. S. Abreu, E. F. Coutinho, and C. I. M. Bezerra, "A blockchain-based architecture for query and registration of Student degree certificates," in *Proc. 14th Brazilian Symp. Softw. Compon., Architectures, Reuse*, Oct. 2020, pp. 151–160, doi: 10.1145/3425269.3425285.

[56] P. Schmidt. (2016). *Blockcerts—An Open Infrastructure for Academic Credentials on the Blockchain*. MIT Media Lab. Accessed: Nov. 21, 2022. [Online]. Available: https://medium.com/mit-media-lab/blockcerts-an-open-infstructure-for-academic-credentials-on-the-blockchain-899a6b880b

[57] Univ. Nicosia. (2017). *University of Nicosia is the First University in the World to Publish Diplomas of All Graduating Students on the Blockchain*. Accessed: Nov. 21, 2022. [Online]. Available: https://www.unic.ac./university-of-nicosia-is-the-first-university-in-the-world-to-publisdiplomas-of-all-graduating-students-on-the-blockchain/

[58] M. Turkanovic, M. Hölbl, K. Košic, M. Hericko, and A. Kamišalic, "EduCTX: A blockchain-based higher education credit platform," *IEEE Access*, vol. 6, pp. 5112–5127, 2018, doi: 10.1109/ACCESS.2018.2789929.

[59] A. M. Antonopoulos and G. Wood, *Mastering Ethereum: Building Smart Contracts and DApps*, 1st ed., Sebastopol, CA, USA: O'Reilly Media, 2019.

[60] A. Preukschat and D. Reed, *Self-Sovereign Identity: Decentralized Digital Identity and Verifiable Credentials*. Shelter Island, NY, USA: Manning, 2021.

[61] D. T. Serranito, "A blockchain-based platform for sharing and verifying education certificates," M.S. thesis, Dept. Comput. Sci. Eng., Univ. Lisbon, Instituto Superior Técnico, Lisbon, Portugal, 2020.

[62] O. S. Saleh, O. Ghazali, and N. B. Idris, "A new privacy-preserving protocol for academic certificates on hyperledger fabric," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 2, pp. 595–609, 2023, doi: 10.14569/ijacsa.2023.0140271.

[63] L. M. Palma, M. A. G. Vigil, F. L. Pereira, and J. E. Martina, "Blockchain and smart contracts for higher education registry in Brazil," *Int. J. Netw. Manage.*, vol. 29, no. 3, pp. 1–21, May 2019, doi: 10.1002/nem.2061.

[64] (1978). *The National Education Act*. [Online]. Available: https://www.parliament.go.tz/polis/uploads/bills/acts/15661476-The%20National%20Education%20Act,%201978.pdf

[65] (2019). *The National Examination Council of Tanzania Act*. Accessed: Jan. 12, 2023. [Online]. Available: https://www.necta.go.tz/exam_rmarts/NECTA_ACT_RE_2019.pdf

[66] (2021). *The National Council for Technical Education Act*. Accessed: Apr. 16, 2024. [Online]. Available: https://www.nacte.go.tz/-content/uploads/2019/01/NACTE-ACT-CAP.-129-No.-9-of-199pdf

[67] Tanzania Commission for Universities (TCU), The Universities Act, 2005. https://www.tcu.go.tz/sites/default/files/iversityAct.pdf. Accessed on: Feb. 11, 2024.

[68] (2006). *The Vocational Education and Training Authority Act*. Accessed: Oct. 20, 2024. [Online]. Available: https://procedures.tic.go./media/TheVocationalEducationandTraining(VETA)ACT.pdf

[69] O. Dib, K.-L. Brousmiche, A. Durand, E. Thea, and E. B. Hamida, "Consortium blockchains: Overview, applications and challenges," *Int. J. Adv. Telecommun.*, vol. 11, nos. 1–2, pp. 51–64, Jan. 2018.

[70] P. Mukherjee and C. Pradhan, "Blockchain 1.0 to blockchain 4.0—The evolutionary transformation of blockchain technology," in *Blockchain Technology: Applications and Challenges* (Intelligent Systems Reference Library), vol. 203, 1st ed., S. K. Panda, A. K. Jena, S. K. Swain, and S. C. Satapathy, Eds., Basel, Switzerland: Springer, 2021, pp. 29–49, doi: 10.1007/978-3-030-69395-4_3.

[71] R. Paulavicius, S. Grigaitis, and E. Filatovas, "A systematic review and empirical analysis of blockchain simulators," *IEEE Access*, vol. 9, pp. 38010–38028, 2021, doi: 10.1109/ACCESS.2021.3063324.

[72] D. Ceke, S. Kunosic, and N. Buzadija, "Smart contract execution costs optimisation on blockchain network," in *Proc. 45th Jubilee Int. Conv. Inf., Commun. Electron. Technol. (MIPRO)*, Opatija, Croatia, May 2022, pp. 1442–1447, doi: 10.23919/MIPRO55190.2022.9803805.

[73] G. Chen, B. Xu, M. Lu, and N.-S. Chen, "Exploring blockchain technology and its potential applications for education," *Smart Learn. Environments*, vol. 5, no. 1, pp. 1–10, Dec. 2018, doi: 10.1186/s40561-017-0050-x.

[74] X. Xu, I. Weber, M. Staples, L. Zhu, J. Bosch, L. Bass, C. Pautasso, and P. Rimba, "A taxonomy of blockchain-based systems for architecture design," in *Proc. IEEE Int. Conf. Softw. Archit. (ICSA)*, Gothenburg, Sweden, Apr. 2017, pp. 243–252, doi: 10.1109/ICSA.2017.33.

[75] S. N. Khan, F. Loukil, C. Ghedira-Guegan, E. Benkhelifa, and A. Bani-Hani, "Blockchain smart contracts: Applications, challenges, and future trends," *Peer-Peer Netw. Appl.*, vol. 14, no. 5, pp. 2901–2925, Apr. 2021, doi: 10.1007/s12083-021-01127-0.

[76] M. Aamir, R. Qureshi, F. A. Khan, and M. Huzaifa, "Blockchain based academic records verification in smart cities," *Wireless Pers. Commun.*, vol. 113, no. 3, pp. 1397–1406, Mar. 2020, doi: 10.1007/s11277-020-07226-0.

[77] E. Leka and B. Selimi, "BCERT—A decentralized academic certificate system distribution using blockchain technology," *Int. J. Inf. Technol. Secur.*, vol. 12, no. 4, pp. 103–118, Nov. 2020. [Online]. Available: https://www.researchgate.net/publication/346487511

[78] B. M. Nguyen, T. C. Dao, and B. L. Do, "Towards a blockchain-based certificate authentication system in Vietnam," *PeerJ Comput. Sci.*, vol. 388, no. 3, pp. 1–14, Mar. 2020, doi: 10.7717/peerj-cs.266.

[79] C. Fan, S. Ghaemi, H. Khazaei, and P. Musilek, "Performance evaluation of blockchain systems: A systematic survey," *IEEE Access*, vol. 8, pp. 126927–126950, 2020, doi: 10.1109/ACCESS.2020.3006078.

[80] J. R. Douceur, "The Sybil attack," in *Proc. 1st Int. Workshop Peer-Peer Syst.*, vol. 2429, P. Drusche, F. Kaashoek, and A. Rowstron, Eds., Berlin, Germany: Springer, Jan. 2002, pp. 251–260, doi: 10.1007/3-540-45748-8_24.

[81] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, Jul. 1982, doi: 10.1145/357172.357176.

[82] S. Peyrott. (2017). *An Introduction to Ethereum and Smart Contracts: A Programmable Blockchain*. Accessed: Nov. 23, 2023. [Online]. Available: https://auth0.com/blog/an-introduction-to-ethereum-and-smart-contracts-part-2/

[83] G. Wood. (2014). *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. Accessed: Feb. 23, 2023. [Online]. Available: https://ethereum.github.io/yellowper/paper.pdf

[84] Binance. *BNB Smart Chain (BSC)*. Accessed: Apr. 26, 2024. [Online]. Available: https://w.bnbchain.org/en/bnb-smart-chain

[85] S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[86] V. Buterin. (2014). *Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform*. [Online]. Available: https://ethereum.org/669c9e2e2027310b6b3cdce1c52962/Ethereum_Whitepaper_-_Buterin_2014.pdf

[87] E. Elrom, "Hyperledger," in *The Blockchain Developer*. CA, USA: Apress, 2019, pp. 299–348, doi: 10.1007/978-1-4842-4847-8_8.

[88] C. Lee. (2011). *Litecoin—Open Source P2P Digital Currency*. Accessed: Aug. 24, 2023. [Online]. Available: https://litecoin.org/

[89] D. Schwartz, N. Youngs, and A. Britto. (2014). *The Ripple Protocol Consensus Algorithm*. Accessed: Jul. 20, 2023. [Online]. Available: https://ripple.com/

[90] E. Harris-Braun, N. Luck, and A. Brock. (2018). *Holochain: Scalable Agent-centric Distributed Computing*. Accessed: Aug. 24, 2023. [Online]. Available: https://thub.com/holochain/holochain-proto/blob/whitepaper/holochain.pd

[91] P. J. Windley. (Oct. 3, 2016). *How Sovrin Works: A Technical Guide From the Sovrin Foundation*. Accessed: Mar. 29, 2023. [Online]. Available: https://sovrin.g/library/how-sovrin-works/

[92] R. G. Brown, J. Carlyle, I. Grigg, and M. Hearn, "Corda: An introduction," R3, New York, NY, USA, Tech. Rep., 2016, doi: 10.13140/RG.2.2.30487.37284.

[93] X. Lu, "Design and features of block chain and algorithm," in *Proc. 3rd Int. Conf. Big Data Economy and Inf. Manag. (BDEIM)*, 2023, pp. 562–569, doi: 10.2991/978-94-6463-124-1_66.

[94] H. Taherdoost, "Smart contracts in blockchain technology: A critical review," *Information*, vol. 14, no. 2, p. 117, Feb. 2023, doi: 10.3390/info14020117.

[95] N. Szabo, "Formalizing and securing relationships on public networks," *1st Monday*, vol. 2, no. 9, pp. 1–21, Sep. 1997, doi: 10.5210/fm.v2i9.548.

[96] M. M. Rahman, M. T. Kabir Tonmoy, S. R. Shihab, and R. Farhana, "Blockchain-based certificate authentication system with enabling correction," 2023, *arXiv:2302.03877*.

[97] M. Alizadeh, K. Andersson, and O. Schelén, "Efficient decentralized data storage based on public blockchain and IPFS," in *Proc. IEEE Asia–Pacific Conf. Comput. Sci. Data Eng. (CSDE)*, Dec. 2020, pp. 1–8, doi: 10.1109/CSDE50874.2020.9411599.

[98] J. Benet, "IPFS–content addressed, versioned, P2P file system," 2014, *arXiv:1407.3561*.

[99] J. Jayabalan and N. Jeyanthi, "Scalable blockchain model using off-chain IPFS storage for healthcare data security and privacy," *J. Parallel Distrib. Comput.*, vol. 164, pp. 152–167, Mar. 2022, doi: 10.1016/j.jpdc.2022.03.009.

[100] R. Shi, R. Cheng, B. Han, Y. Cheng, and S. Chen, "A closer look into IPFS: Accessibility, content, and performance," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 8, no. 2, pp. 1–31, May 2024, doi: 10.1145/3656015.

[101] M. H. Mughal, Z. A. Shaikh, K. Ali, S. Ali, and S. Hassan, "IPFS and blockchain based reliability and availability improvement for integrated rivers' streamflow data," *IEEE Access*, vol. 10, pp. 61101–61123, 2022, doi: 10.1109/ACCESS.2022.3178728.

[102] Open Badges. *Open Badges V2.0 IMS Final Release*. Accessed: Oct. 4, 2023. [Online]. Available: https://www.imsglobal.org/sites/default/files/Badges/OBp0Final/index.html

[103] K. Sansone. (2019). *Malta is First Country to Put Education Certificates on Blockchain*. Maltatoday. Accessed: Nov. 21, 2022. [Online]. Available: https://www.maltoday.com.mt/news/national/93148/malta_is_first_country_to_put_education_certificates_on_blockchain#.Y3s2YUxBzDc

[104] Li Zhao. (2018). *Blockchain Based Academic Certificate Authentication System Overview*. Univ. Burmigham. Accessed: Feb. 29, 2024. [Online]. Available: https://blog.bham.ac.uk/itinnovation/2018/05/24/blockchn-based-academic-certificate-authentication-system-overview/

[105] W. Gräther, S. Kolvenbach, R. Ruland, J. Schütte, C. F. Torres, and F. Wendland, "Blockchain for education: Lifelong learning passport," in *Proc. 1st ERCIM Blockchain Workshop*, 2018, pp. 1–8, doi: 10.18420/blockchain2018.

[106] Remix Project. *Remix IDE (Version 0.37.3)*. Ethereum Found. Accessed: Nov. 15, 2023. [Online]. Available: https://remix.ethereum.org/#lang=en&optize=false&runs=200&evmVersion=null&version=soljsonv0.8.22ommit.4fc1097e.js

[107] Microsoft. (2015). *Visual Studio Code*. Accessed: Nov. 27, 2023. [Online]. Available: https://code.visualstudio.com/

[108] React. (2013). *React (Version 18.2.0)*. Facebook. Accessed: Nov. 14, 2023. [Online]. Available: https://github.com/facebook/react

[109] Node.js. (2009). *Node.Js (Version 20.9.0)*. OpenJS Found. Accessed: Nov. 15, 2023. [Online]. Available: https://nodejs.org/en

[110] Ethereum. (2014). *Web3.js (Version 4.2.2)*. Ethereum Found. Accessed: Nov. 15, 2023. [Online]. Available: https://github.com/web3/web3.js

[111] Ethers.js contributors. (2016). *Ethers.Js (Version 6.8.2)*. GitHub. Accessed: Nov. 26, 2023. [Online]. Available: https://github.com/ethers-io/ethers.js
[112] MetaMask. (2016). *Metamask-Extension*. GitHub. Accessed: Nov. 22, 2023. [Online]. Available: https://github.com/MetaMask/metamask-extension/releases
[113] Ethereum. (2015). *Go-Ethereum: Official Go Implementation of the Ethereu Protocol*. GitHub. Accessed: Nov. 23, 2023. [Online]. Available: https://github.com/hereum/go-ethereum
[114] ConsenSys. (2016). *Infura*. Accessed: Dec. 4, 2023. [Online]. Available: https://wwinfura.io/
[115] (2023). *Protocol Labs*. Accessed: Apr. 28, 2024. [Online]. Available: https://github.com/ipfs-inactive/js-ipfs-http-clien
[116] Binance. *BSC Testnet | BNB Chain Documentation*. Testnet. Accessed: Dec. 5, 2023. [Online]. Available: https://docs.bnbchain.org/docs/Ctestnet
[117] Off. J. Eur. Union. (2016). *European Parliament & Council, Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the Protection of Natural Persons With Regard to the Processing of Personal Data and on the Free Movement of Such Data (General Data Protection Regulation)*. Accessed: Jun. 17, 2024. [Online]. Available: https://eur-lex.europa.eu/i/reg/2016/679/oj/eng
[118] California Legislative Inf. (2018). *California Consumer Privacy Act of 2018*. Accessed: Jan. 20, 2024. [Online]. Available: https://oag.ca.gov/privacy/ccpa#:~:text=TheCaliforniaConsumerPrivacyAct,howtoimplementthelaw
[119] Gazette United Republic Tanzania. (2023). *The Personal Data Protection (Personal Data Collection and Processing) Regulations*. Accessed: Feb. 19, 2024. [Online]. Available: https://www.mawasiliano.go.tz/uploads/documents/sw-1691159153-GNNO.449COF2023.pdf
[120] E. Zhou, S. Hua, B. Pi, J. Sun, Y. Nomura, K. Yamashita, and H. Kurihara, "Security assurance for smart contract," in *Proc. 9th IFIP Int. Conf. New Technol., Mobility Secur. (NTMS)*, Feb. 2018, pp. 1–5, doi: 10.1109/NTMS.2018.8328743.
[121] X. Wu, Z. Zou, and D. Song, *Learn Ethereum: Build Your Own Decentralized Applications With Ethereum and Smart Contracts*, 1st ed., Birmingham, U.K.: Packt, 2019.
[122] Tanzania Commission Universities. (2023). *Vital Stats on University Education in Tanzania 2022*. Accessed: Dec. 6, 2023. [Online]. Available: https://www.u.go.tz/publications/higher-education-statistics
[123] NACTVET, (2024). *Registered and Accredited Institutions* Accessed: Apr. 4, 2024. [Online]. Available: https://nactvet.go.tz/registered-institutions
[124] NACTVET. (2021). *TVET Indicators Report*. Accessed: Sep. 9, 2024. [Online]. Available: https://www.nactvet.go.tz/page/reports

SAID H. SAID received the B.Sc. degree in information technology from the University of Iringa (UoI), Tanzania, in 2011, and the M.Sc. degree in computer science from the University of Dodoma (UDOM), Tanzania, in 2015. He is currently pursuing the Ph.D. degree in information and communication science and engineering with the School of Computational and Communication Science and Engineering (CoCSE), The Nelson Mandela African Institution of Science and Technology (NM-AIST), Tanzania. He has ten years of teaching experience with the University of Dodoma, Tanzania, where he is currently an Assistant Lecturer. His research interests include blockchain technology and artificial intelligence applications. He has published in the field of blockchain-enabled certification, verification, and forgery prevention.

RAMADHANI S. SINDE (Member, IEEE) received the B.Sc. and M.Sc. degrees in engineering and technologies in telecommunication from Moscow Technical University of Communication and Informatics, the master's Diploma degree in wireless and mobile computing from the Center for Development of Advanced Computing, India, and the Ph.D. degree in information and communication science and engineering from The Nelson Mandela African Institution of Science and Technology (NM-AIST), Tanzania. He is currently a Lecturer with the School of Computational and Communication Science and Engineering (CoCSE), NM-AIST. He specializes in electronics and telecommunication engineering. He has authored or co-authored more than 20 papers in internationally refereed journals and conferences. His research interests include telecommunications and informatics, wireless and mobile communication, wireless sensor networks, the Internet of Things, and embedded systems.

EFRAIM M. KOSIA received the B.Sc. degree in agriculture general and the M.Sc. degree in soil science and land management from the Sokoine University of Agriculture (SUA), Tanzania, in 1998 and 2003, respectively, the M.Sc. degree in educational science and technology (educational management, evaluation, and assessment) from the University of Twente, The Netherlands, in 2010, and the Ph.D. degree in educational assessment and evaluation from Mwenge Catholic University (MWECAU), Tanzania, in 2018. He is currently the Director of Academic Affairs with The Nelson Mandela African Institution of Science and Technology (NM-AIST), Tanzania. His research interests include assessment and evaluation, quality assurance, educational management, and curricula studies.

MUSSA A. DIDA received the B.Sc. degree in computer engineering and information technology from the University of Dar es Salaam (UDSM), Tanzania, in 2008, the M.Sc. degree in telecommunication engineering from the University of Dodoma (UDOM), Tanzania, in 2011, and the Ph.D. degree in information and communication engineering from Beijing Institute of Technology (BIT), China, in 2017. He is currently a Senior Lecturer and the Dean of the School of Computational and Communication Science and Engineering (CoCSE), The Nelson Mandela African Institution of Science and Technology (NM-AIST), Tanzania. He is also a Deputy Leader of the Centre of Excellence for ICT in East Africa (CENIT@EA). He has successfully supervised/co-supervised more than 30 M.Sc. students and more than seven Ph.D. students to graduation. He has published more than 70 articles in highly reputable journals. His research interests include digital signal processing, fractional Fourier transform signals and systems, physical layer security, multiple input multiple output (MIMO) antennas, and wireless communication systems.