

Redefining Governmental Services Through Blockchain and Smart Contracts

Ibrahim Ramadan Abdelhamid^{1*}, Islam Tharwat Abdel Halim^{2,3}, Ibrahim Abdelmoniem Ibrahim¹, Abd El-Majeed Amin Ali¹



¹ Faculty of Computers and Information, Minia University, Minia 61519, Egypt

² School of Information Technology and Computer Science (ITCS), Nile University (NU), Giza 12677, Egypt

³ Center for Informatics Science (CIS), Nile University, Sheikh Zayed 12677, Egypt

Corresponding Author Email: Ibrahim.ramadan2207@gmail.com

<https://doi.org/10.18280/mmep.100503>

Received: 12 August 2023

Revised: 13 September 2023

Accepted: 19 September 2023

Available online: 27 October 2023

Keywords:

blockchain, Ethereum, smart contract, e-government, MetaMask, Interplanetary File System, solidity

ABSTRACT

This study explores the potential of blockchain technology to redefine public administration, focusing on the integration of Ethereum, a blockchain platform, and the Interplanetary File System (IPFS) for notarial certification issuance. The core aim is to evaluate the capacity of this technology to augment governmental efficacy, ensure transparency in service provision, decentralize data management, and maintain information integrity. The architectural components of the system comprise Ethereum's smart contracts, Ether, gas, and a decentralized application, supplemented by IPFS as a decentralized file storage system for a secure and transparent certificate issuance mechanism. Scalability assessments indicated efficient processing of multiple transactions per second (TPS), suggesting the system's capability to service a considerable number of simultaneous users. The encryption and decryption performance exhibited by IPFS, particularly for small content sizes of 250 KB and 500 KB, was near-instantaneous. Average times for deployment and execution were recorded as 9 seconds and 6 seconds, respectively. In conclusion, the synergistic integration of Ethereum and IPFS exhibits the potential to significantly transform public administration by augmenting efficiency and transparency in the delivery of citizen-centric services. Notably, the incorporation of IPFS for secure file storage and hash transmissions was instrumental in optimizing cost-efficiency.

1. INTRODUCTION

Globally, governments are central to the provision of essential services within their jurisdictions, including but not limited to infrastructure, roadways, seaports, and airports [1]. Further, they interact with their citizens through the provision of licenses, regulatory frameworks, and financial operations, thereby positioning themselves as service providers. In an era of accelerating technological transformation, governments worldwide are increasingly adopting intelligent systems to reshape their bureaucratic processes [2], with a primary focus on enhancing operational efficiency. It is recognized that efficiency in government operations is not merely an administrative convenience but a factor that directly impacts improved public services, heightened citizen trust, streamlined resource allocation, and significant cost savings [3]. It is widely accepted that effective governance, supported by evidence-based decision-making, positively influences economic growth and societal welfare, thereby fostering a content and sustainable nation [4].

However, previous studies have indicated that achieving optimal administrative efficiency in government is fraught with significant challenges. Factors such as bureaucratic inertia and resistance to change can pose substantial obstacles, hindering efforts to streamline workflows and introduce innovative practices. Moreover, the evolving digital landscape introduces increased complexity in ensuring data security and privacy, thereby emphasizing the need for robust measures [5].

The secure handling of sensitive information, whether personal, financial, or classified, is of paramount importance, as vulnerabilities in government systems can erode citizens' trust through unauthorized access, misuse, or data leakage. Further, the complex network of government departments, characterized by siloed structures and differing priorities, presents coordination challenges.

The digitization of services, such as online portals for citizen interactions and electronic document management systems, has proven effective in reducing paperwork and manual tasks, resulting in faster and more accurate administrative processes. Furthermore, decentralized technologies like blockchain are emerging as a viable alternative that governments can leverage for digital service delivery [6]. The benefits of blockchain lie in its inherent ability to avoid centralized authority while ensuring data integrity and immutability. The last decade has seen a rapid rise in these distributed decentralized networks, primarily due to their ability to guarantee openness without compromising resilience, privacy, and security. In essence, the use of these systems has shown significant promise in contexts where a large, decentralized crowd, such as a country's citizens, is involved [6].

In this study, we propose the use of blockchain technology to optimize government administrative efficiency. The primary aim was to understand how blockchain technology can be effectively employed to enhance public service delivery and management. The practicality, utility, and process of

integrating IPFS (a decentralized file storage system) and Ethereum (a smart contract-enabled blockchain) for issuing birth certificates through the Notarial office were investigated.

Our findings illustrate how this technology can transform public administration by providing transparent, trustworthy services. By decentralizing data management, ensuring information integrity and immutability, and eliminating reliance on centralized authorities, this approach demonstrated the potential to revolutionize interactions between citizens, businesses, and government entities.

2. RELATED WORK

As the exploration of blockchain technology's application within governmental administrative services is undertaken, it becomes imperative to delve into the existing scholarly discourse in this domain. Prior research illuminates various dimensions of employing blockchain to augment governmental operations and service provisioning. This review furnishes a contextual landscape for the present study and pinpoints the lacunae that our research endeavours to address.

A recent survey, conducted by our team, scrutinized the ongoing projects and use cases involving blockchain technology endorsed by global governments [7]. An upsurge in the extent of discourse involving governmental applications of blockchain underscores the mounting acknowledgment of blockchain's potential value in governance. Governmental entities across the globe have embarked on exploratory endeavours to comprehend how this technology might enhance the public sector.

Despite burgeoning enthusiasm, it is crucial to acknowledge that the practical implementation of blockchain technology in governance remains predominantly in the experimental phase. Diverse governmental bodies have instigated trials and projects to evaluate its feasibility and efficacy across varied functions and services. These nascent efforts underscore that blockchain is still in its evolutionary phase within governmental affairs. However, the potential impact of blockchain is significant, presenting an opportunity to establish robust, transparent, and efficient mechanisms for managing public data, overseeing public finances, and delivering essential services [7].

Hou embarked on an exploration into the application of blockchain technology within the context of e-government, with a specific focus on its implementation in China [8]. A detailed analysis of the Comprehensive Experimental Area of Big Data project, initiated by the Chancheng District in Guangdong Province in 2016, was undertaken. The study evaluated the potential impact of blockchain integration on Chinese e-government practices, taking into account its framework, impediments, and potential advantages. The Chancheng District, in association with 21ViaNet China Inc., established an e-government platform powered by blockchain. This platform aimed to address challenges concerning identity verification, credit assessment, and transparent information sharing.

The primary objective was to leverage the capabilities of blockchain in ensuring authenticity, security, and transparency, which could in turn provide solutions to complex challenges faced by China's e-government ecosystem. This initiative also sought to foster trust among governmental entities, enterprises, and the general public. Despite these aspirations, the study

found that challenges such as platform costs, record preservation, security, and management distribution emerged [8].

In another seminal study, Khan et al. delved into the realm of blockchain technology's application within governmental operations, concentrating specifically on the United Arab Emirates (UAE), and more precisely, a governmental entity located in Dubai [9]. The driving aim of this research was to investigate the potential of blockchain to seamlessly integrate e-business and e-government services. The aspiration was to bolster process efficiency, fortify security measures, and ensure robust data synchronization.

The investigators proposed a consortium blockchain framework purpose-built for a Unified Corporate Registry. This framework, developed using the Hyperledger Fabric platform, was designed to establish connectivity with peripheral nodes, encompassing both public registries and businesses [9]. Its core functionality was to streamline the dissemination of license information amongst various stakeholders. It is noteworthy that the membership structure of this blockchain ecosystem was clearly defined, incorporating critical roles such as data publishers, subscribers, service providers, and index managers. This diverse array of participants constituted a cohesive network, the goal of which was to facilitate seamless data exchange and management within the registry.

Despite the potential of this approach, Khan et al. also acknowledged certain challenges, including issues pertaining to scalability and regulatory compliance. Nonetheless, the potential benefits were substantial, with the framework offering the prospect of significantly expedited processes and enhanced data-sharing capabilities.

In a separate study, Pinter et al. put forth an innovative approach aimed at enhancing e-government services, with a specific emphasis on fortifying e-ID systems via the integration of blockchain technology [10]. This novel decentralized framework, in contrast to the traditional centralized authentication model, not only significantly bolstered security against potential cyberattacks but also addressed concerns pertaining to data protection in the context of public blockchain storage. To ensure data privacy, the proposal advocated for a hybrid strategy wherein technical references were stored on the blockchain, while user data was securely housed in localized repositories. This privacy-centric architectural approach allowed for the accommodation of multiple user identities, each verifiable through the digital imprints of public keys.

The architecture was founded on the SVN-G draft law, which facilitated robust user identification processes. Operationally, this entailed users selecting authorized Know Your Customer (KYC) providers via an ID portal for identity validation [11]. Upon successful KYC verification, user data was securely recorded on the public blockchain, with a KYC-generated signature enabling data access without disclosing personal information, thereby establishing trust between users and KYC providers. Additionally, offline data, such as ID numbers linked to public keys, was securely maintained by KYC providers, serving as a cooperative resource for authorities in cases involving illicit activities. Notably, the blockchain implementation proved highly effective in thwarting Denial-of-Service (DoS) attacks, underscoring its transformative role in streamlining and securing the KYC process, which itself is a security tool.

In an expansive survey conducted by Negara et al. [12], an

investigation into the transformative impact of blockchain technology on government agencies was conducted, particularly focusing on the capabilities of smart contracts. The study illuminated the revolutionary nature of blockchain's decentralized approach, which has profound implications for data transactions and communication within databases. Prominently, blockchain technology has been identified as offering effective solutions to pressing issues such as security, privacy, and traceability. One of the significant findings from this research was the widespread adoption of blockchain across diverse sectors, with a specific emphasis on its role in enhancing e-government services and improving overall operational efficiency. Smart contracts, which autonomously execute predefined terms and agreements, have been instrumental in streamlining processes by eliminating intermediaries and automating the fulfillment of contractual conditions.

Despite these substantial benefits, Negara et al. also identified challenges associated with the adoption of blockchain technology and smart contracts, including vulnerabilities to cyberattacks and potential disruptions to traditional roles due to disintermediation. However, it is vital to note that the advantages, such as reduced costs, enhanced transparency, and increased trust in governance, far outweigh these challenges.

Another significant endeavor, the Transparency Project, was conducted in collaboration with the Inter-American Development Bank and the Office of the Inspector General of Colombia [13]. This initiative aimed to confront corruption within government processes prone to malfeasance, by integrating blockchain and distributed ledger technologies, thereby fostering transparency and accountability. The project concentrated its efforts on leveraging the Ethereum blockchain, where it successfully devised a proof-of-concept (PoC) software solution, whose core objectives encompassed the prevention of record tampering, the facilitation of transparent transactions, and the automation of operations through smart contracts.

Notwithstanding, the Transparency Project underscored the necessity for more extensive cultural and societal transformations, vital shifts to effectively counter deep-rooted and systemic corruption practices, which often extend beyond technological solutions alone.

While various government projects have leveraged blockchain technology to administer and improve the efficiency of services in the past, specific challenges have emerged from these attempts. These shall form the basis of further discussion and analysis in this study.

The potential of blockchain technology in enhancing administrative government services is undeniably substantial, yet it is accompanied by significant challenges and research gaps that warrant attention. Paramount among these is the question of cost-effectiveness. Governments, particularly those operating on limited budgets, are confronted with the substantial expenses associated with the implementation of blockchain technology [14]. Additionally, security, while a cornerstone of blockchain, remains susceptible to vulnerabilities, necessitating continuous investigation to identify and mitigate potential risks inherent in governmental data and transactions. Concurrently, concerns related to scalability persist, especially when considering high-volume government service transactions. This scenario underscores the necessity for innovative solutions to ensure the smooth operation of services [15].

Moreover, the delicate balance between data privacy and transparency in government services necessitates exploration of blockchain architectures that could potentially bridge this gap. The research approach proposed in this paper extends the application of blockchain's fundamental attributes, such as immutability, transparency, and security, to revolutionize the delivery of government services through the seamless integration of the InterPlanetary File System (IPFS) [16]. This integration plays a pivotal role in modernizing governmental operations and addressing a multitude of challenges, including the secure and permanent preservation of critical government records.

By leveraging IPFS, documents and data are distributed across a decentralized network of nodes, thereby enhancing resilience and thwarting any attempts at tampering. The decentralized nature of IPFS bolsters security by eliminating single points of vulnerability. Ethereum smart contracts complement this by establishing timestamped, immutable records of transactions and interactions with these documents, thus guaranteeing the durable preservation of historical data. Once deployed, smart contracts remain impervious to alterations, and their logic controls access permissions with precision.

Beyond security, the technologies related to blockchain and smart contracts excel in efficiency, redistributing management responsibilities, automating processes, and reducing reliance on intermediaries. IPFS supplements this by ensuring that all relevant parties have access to documents without compromising data integrity. Furthermore, the inherent transparency in Ethereum's public blockchain and smart contracts creates an auditable and transparent trail of every transaction and interaction, fostering trust among citizens and authorities alike. This approach potentially obviates the need for costly centralized data storage solutions, resulting in significant savings in government operations.

3. PRELIMINARIES

3.1 Blockchain concepts

Blockchain technology underpins our research:

Decentralization denotes the dispersion of control, authority, and decision-making across a network of participants, as opposed to relying on a central entity [17]. Within the context of blockchain and distributed systems, decentralization serves to enhance transparency, security, and resilience.

Permissioned and Permissionless Blockchains: Two key categories of blockchains exist: permissioned and permissionless. Permissioned blockchains restrict access to authorized users, ensuring tighter controls. In contrast, permissionless systems are open for participation by anyone, promoting inclusivity [18].

Our research leverages the Ethereum blockchain, which offers both permissioned and permissionless modes.

3.2 Why Ethereum for dApps?

The choice of Ethereum as the preferred blockchain for this research is driven by its unique advantages over other blockchain systems:

Smart Contract Execution: Ethereum boasts robust and versatile execution environment known as the

Ethereum Virtual Machine (EVM). The EVM ensures uniform code execution across all network nodes, fostering trust, consensus, and security [19]. This makes it ideal for the creation of decentralized applications (dApps).

Smart Contracts: Ethereum's support for smart contracts, self-executing agreements embedded in code, enables automated execution of transactions and agreements without intermediaries [19]. These contracts underpin various dApps functionalities.

Solidity: Ethereum's primary programming language, Solidity, facilitates the creation of smart contracts [20]. Drawing inspiration from C++, Solidity offers features like inheritance and code libraries for reusable code. This empowers developers to craft secure and efficient dApps with modular structures, promoting code reusability.

Cryptocurrency Wallets and Gas: To interact with dApps, users rely on cryptocurrency wallets like MetaMask, which also securely store private keys, ensuring user control over their assets and data. Additionally, Ethereum's concept of gas measures the computational cost of actions on its blockchain, providing fair compensation for computational resources and incentivizing efficient network use.

3.3 Distributed storage options

IPFS is a protocol and peer-to-peer distributed network designed to reshape file systems in a distributed landscape [21]. By using content addressing through cryptographic hash functions, IPFS assigns unique identifiers to data, enabling deduplication and efficient caching. This innovative approach ensures that identical content consistently retains the same identifier, bolstering reliability and making data resilient to network challenges.

Functioning as a decentralized cloud storage system, StorJ enables users to lease their untapped storage capacity to the network [22]. Through blockchain technology, StorJ ensures data security, integrity, and privacy by encrypting files into smaller fragments distributed across network nodes, minimizing vulnerabilities and single points of failure. Users have granular control over their data, managing encryption keys and access permissions.

3.4 Decentralized application

An integrated development environment (IDE) called Remix simplifies the creation, testing, and deployment of smart contracts on the Ethereum network [23]. Its user-friendly graphical interfaces and plugins enhance the development lifecycle and serve as the portal via which the user can interact with a blockchain-based system.

Overall, the system proposed in this research would work such that, a user logs into the dApp (whose core logic is implemented using smart contracts written in Solidity) using their MetaMask wallet. Next, they upload a file through the dApp's user interface provided by Remix for decentralized and distributed storage on IPFS. Remix then communicates with the Ethereum blockchain to create a new smart contract instance that records information about the uploaded file, including its IPFS hash and access conditions.

Other users can access the file by interacting with the smart contract. The smart contract verifies their permissions and provides them with the IPFS link to retrieve the file. Finally, when users download the file, they retrieve it from the IPFS network using the provided IPFS hash.

In this way, Ethereum's smart contracts handle access

control and ownership of files, while IPFS ensures decentralized and efficient storage of the files. The user interface (Remix) and the cryptocurrency wallet (MetaMask) facilitate user interactions and transactions with the dApp. These technologies come together to create a decentralized file-sharing dApp with trust, transparency, and user control.

4. PROPOSED APPROACH

4.1 Architecture

In this section, we will provide a detailed explanation of the proposed system architecture, the setup of the Ethereum platform, and the various functionalities it offers. The content will encompass the design and structure of the system, the installation, and configuration of the Ethereum blockchain platform, as well as an exploration of the different capabilities and features provided, including decentralized applications and smart contract functionalities.

4.1.1 System architecture

Components of Ethereum used in the experiment include:

The Ethereum blockchain is a public, decentralized, distributed ledger that maintains a record of all transactions and smart contracts executed on the platform. Serving as the very foundation of the Ethereum network, the blockchain provides the platform for executing applications in a decentralized manner [24]. Ether (ETH) is the native digital token on Ethereum, and its role is multi-faceted. It serves as a reward mechanism for network validators, facilitates fluid value exchange, and offers economic incentives for upholding network security through.

Smart contracts are contracts with the terms of the agreement embedded directly into code, making them self-executing. When these predetermined conditions are fulfilled, these contracts execute and enforce the agreed-upon terms autonomously [25]. For the development of smart contracts, Solidity is a high-level, object-oriented programming language used for creating smart contracts on the Ethereum blockchain designed to execute on the Ethereum Virtual Machine (EVM). Speaking of which, the EVM plays a crucial role in executing smart contracts on the Ethereum blockchain. It operates within a sandboxed and isolated environment where the bytecode of smart contracts is executed, ensuring consistency and determinism of contract execution across all participating nodes.

Gas refers to the computational cost required to perform actions or execute code on Ethereum's blockchain. Gas fees, paid in Ether, are necessary to cover the resources spent on computation, storage, and bandwidth. Validators receive gas fees as compensation for including transactions and executing smart contracts, incentivizing network security, and proper transaction processing [26].

Ethereum clients serve as software implementations, code to parse and verify the blockchain that enables users to interact with the Ethereum network. These clients can be full nodes, storing and validating the entire blockchain, or lightweight clients that interact with full nodes to access blockchain data and execute transactions [27]. Additionally, Ethereum wallets, such as MetaMask, are software applications that empower users to manage their Ethereum accounts, store private keys, and interact with the Ethereum network. These wallets facilitate actions like sending and receiving Ether, interacting with smart contracts, and managing digital assets.

Decentralized Applications (dApps) are distributed software applications on Ethereum (and other smart contract-supported blockchains), leveraging these contracts for their functionality. dApps serve several uses, from financial applications to decentralized exchanges, games, governance platforms, and more [28]. They provide users with direct control over their data and assets while also promoting transparency and resistance to censorship. Various tools and frameworks aid Ethereum application development, including libraries, integrated development environments (IDEs), testing frameworks, and deployment tools. Some popular tools in this category include Truffle, Remix IDE, and Hardhat.

The interplay between the components described above creates a decentralized and programmable blockchain platform on which smart contracts and decentralized applications are executed, and the secure transfer of value and digital assets is assured. Each component contributes to the overall functionality and success of the Ethereum network, fostering a robust and versatile ecosystem for blockchain-based applications and use cases.

The diagram presented in Figure 1 provides an overview of the proposed system's high-level architecture. The interaction with the Ethereum ecosystem commences as a user uploads an item to the Interplanetary File System (IPFS). IPFS, functioning as a decentralized file storage platform [29], facilitates the storage and retrieval of files by utilizing distinct content identifiers, known as hashes.

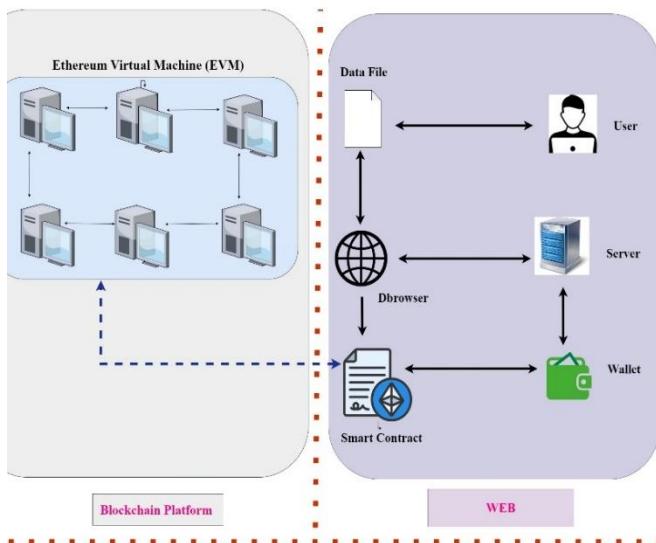


Figure 1. High-level architecture of system

Next, the user engages with a decentralized web browser (Browser), which serves as a gateway to access decentralized applications and services on the Ethereum network. The user provides the necessary parameters, including the generated hash, to the browser, which then acts as an intermediary, transferring the provided parameters to a web server. Essentially, the web server bridges the user and the Ethereum blockchain, as it handles communication and forwards the information to the underlying blockchain platform.

To interact with Ethereum, the user initiates a transaction and transfers an amount of Ether through their wallet - a secure digital storage for the user's private key. The transaction and accompanying data are sent from the user's wallet to Ethereum, specifically to a deployed smart contract. A self-executing program stored on the Ethereum blockchain, a smart contract is programmed to fulfill predefined actions once the specific

conditions are met. The information provided by the user, including the hash and other parameters, is received by the web server. The web server communicates with the Ethereum network, forwarding the information for storage, validation, and execution within the smart contract.

On the Ethereum network, data is stored on the blockchain, utilizing state variables to ensure both immutability and transparency. The network verifies the transaction and carries out the logic embedded within the smart contract that has been deployed.

In a nutshell, the diagram presented in Figure 2 outlines the sequence of steps a user takes when engaging with the Ethereum ecosystem. This journey entails uploading an item onto the IPFS, generating a unique identifier, using a decentralized web browser to relay this data to a web server, making a payment in Ether through a digital wallet, and finally storing, validating, and executing the data on the Ethereum blockchain network via a deployed smart contract. This process exemplifies the decentralized and programmable essence of the Ethereum ecosystem, making it well-suited for secure and transparent interactions.

4.1.2 IPFS

IPFS was chosen as the decentralized file storage solution over other alternatives such as StorJ for several reasons that align with the goals and requirements of our system:

-Content Addressing: IPFS uses content addressing, where each file is identified by a unique cryptographic hash derived from its content. This ensures that identical files will have the same hash, promoting deduplication and efficient data caching. Content addressing also bolsters data integrity and reliability.

-Decentralization and Redundancy: IPFS operates on a peer-to-peer network where files are distributed across multiple nodes. This decentralized architecture enhances data redundancy and availability. Files are not stored on a single server, reducing the risk of data loss due to server failures.

-Resilience to Network Challenges: IPFS is designed to work efficiently even in scenarios with unreliable or intermittent network connectivity. This resilience is crucial for a decentralized system where nodes may join or leave the network at any time.

-User Control: IPFS allows users to retain control over their data. Users can decide when and how to distribute their files on the network, providing them with a level of ownership and privacy that centralized cloud storage solutions may not offer.

-Interoperability: IPFS is designed to work well with blockchain technologies, including Ethereum. The content addressing of files aligns with Ethereum's approach to data storage, making it easier to integrate IPFS with Ethereum-based smart contracts, as seen in the proposed system.

4.1.3 Workflow of the system

Figure 3 shows a step-by-step interaction between the Ethereum blockchain and IPFS to fulfil a request for a birth certificate.

A resident initiates the process by requesting his birth certificate. Instead of using traditional paper-based methods, they opt for the benefits offered by blockchain technology and IPFS to ensure a secure and efficient transfer. First, the resident gathers and uploads all necessary documents needed to obtain the birth certificate and uploads them to the IPFS. To ensure the privacy and security of the documents when in transit on the network, they are encrypted using appropriate techniques.

After uploading the documents to IPFS, the resident

generates a unique cryptographic hash, which represents the encrypted birth certificate. This hash acts as a digital fingerprint, guaranteeing the document's integrity. The resident then sends this hash to Ethereum, and to interact with the blockchain, the resident utilizes Metamask, a digital wallet application. Metamask enables secure management of Ethereum accounts and transaction signing. The resident signs the transaction containing the hash of the birth certificate document using Metamask to demonstrate ownership and authorization. The transaction, including the hashed document, is sent to the Ethereum network and processed by a smart contract. Smart contracts are self-executing agreements encoded on the blockchain with predefined rules. In this case, the smart contract manages specific commands and the flow of information related to birth certificates. Upon executing the necessary commands, the smart contract transfers the encrypted hash from the resident's address to the address of the Notarial office. This transfer is recorded on the blockchain, ensuring transparency and immutability.

Upon receiving the hash on the blockchain, the Notarial office retrieves the encrypted birth certificate document from IPFS using the hash. With access to the necessary decryption keys, they decrypt the document, making it accessible in its original form. After decrypting the birth certificate document, the Notarial office sends the original document to the resident. A secure method, such as IPFS, ensures the safe transfer of the document back to the resident. This ensures that the resident receives the verified and unaltered birth certificate. The

integration of the Ethereum blockchain and IPFS in this process offers several advantages, including the immutability and transparency of the blockchain, the decentralized storage and retrieval of documents on IPFS, and the security provided by encryption and digital signatures. This innovative combination showcases the potential for secure and efficient interactions within the Ethereum ecosystem, especially in scenarios involving sensitive data and official documentation.

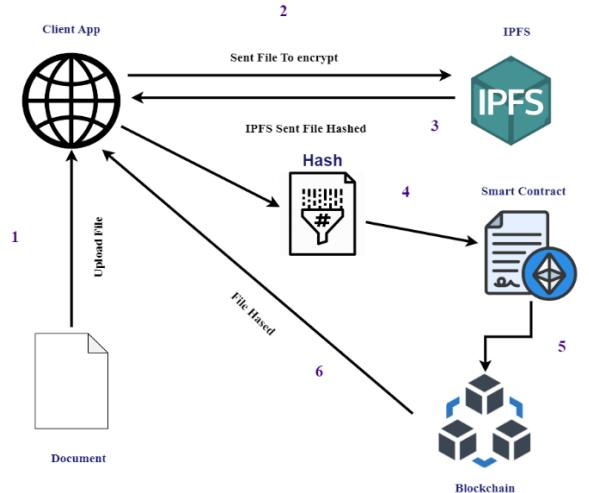


Figure 2. IPFS workflow procedure

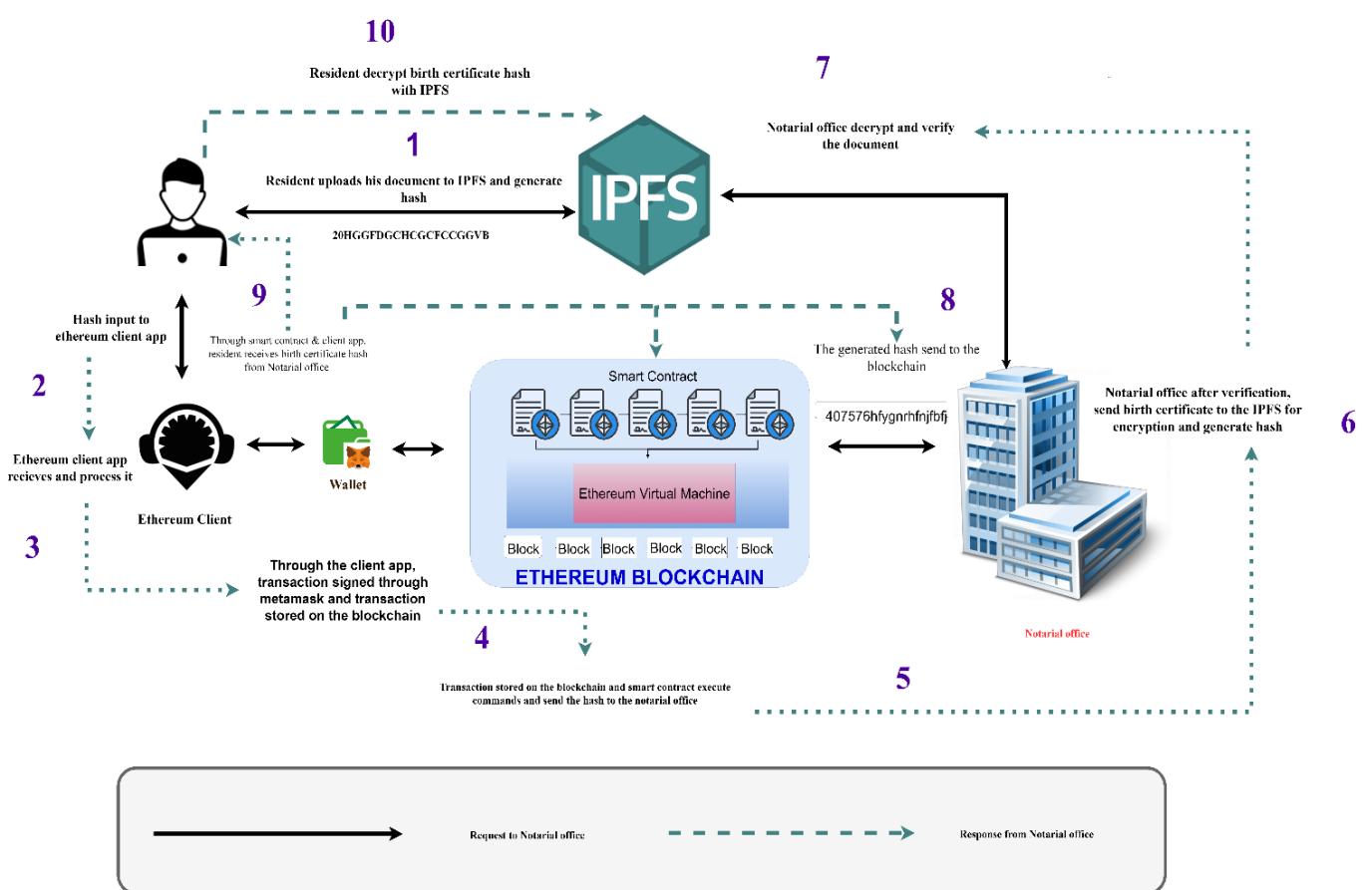


Figure 3. Workflow of system

4.1.4 Functionalities

This section outlines the various functionalities that the proposed system leverages to enable seamless interactions and

operations within the Ethereum ecosystem. Each subsection highlights a specific function for the overall user experience and efficient utilization of the system.

(1) Send validation request

The "Send Validation Request" functionality empowers users to initiate the validation process for specific actions within the system. Whether it involves submitting important data, validating transactions, or confirming identity-related information, this feature allows users to trigger the validation process seamlessly. Once a request is generated, it is propagated to the Ethereum network, triggering relevant smart contracts and initiating the validation workflow.

Algorithm 1 Send Validation Request

```
// Algorithm to send a validation request
function sendValidationRequest(address walletAddress,
string memory cid) external {
// Check if the CID is not empty
if (bytes(cid).length > 0) {
// Check if a validation request has not already been made
for the given walletAddress
if (validationRequests[walletAddress].walletAddress ==
address(0)) {
// Create a new ValidationRequest and store it in the
mapping
validationRequests[walletAddress]=
ValidationRequest(walletAddress, cid, false);
// Emit ValidationRequested event
emit ValidationRequested(walletAddress, cid);
} else {
// Throw an error - Validation already requested
throw "Validation already requested";
}
} else {
// Throw an error - CID cannot be empty
throw "CID cannot be empty";
}
}
```

Let's explore this functionality through a real-world scenario:

Imagine a user, Sarah, is using our decentralized system for conducting a secure and validated transaction on the Ethereum blockchain. In this scenario, Sarah wants to send a significant amount of crypto to another user, John, and she wants to ensure the transaction is validated to minimize the risk of fraudulent activity. She decides to use the "Send Validation Request" feature.

To achieve this, Sarah logs into the dApp and navigates to the "Send Validation Request" section. Here, she specifies her Ethereum wallet address and includes a unique content identifier (CID) associated with the transaction details.

The algorithm behind the "Send Validation Request" functionality performs several tasks:

1. It checks that the CID is not empty, ensuring that Sarah provides relevant transaction information.
2. It verifies if a validation request has not already been made for Sarah's wallet address. This step prevents duplicate requests for the same transaction.
3. If all conditions are met, a new validation request is created and stored in the system, associated with Sarah's wallet address and the provided CID.
4. An event, "ValidationRequested," is emitted, signaling the initiation of the validation process.

In this way, the "Send Validation Request" functionality allows Sarah to trigger the validation workflow for her financial transaction. The CID serves as a unique identifier for her transaction details, and the smart contract ensures that only

one validation request is generated for this specific action. This feature enhances trust and security in the dApp, giving users like Sarah confidence that her transactions are being validated and monitored for added protection against potential fraud.

(2) Send validation completion

The "Send Validation Completion" function enables users to respond to validation requests initiated by other participants within the system. Whether it's a confirmation of received goods, the completion of a particular task, or the approval of a critical transaction, users can efficiently signal the successful validation of an action. This helps streamline processes and ensures the smooth functioning of smart contracts and dApps within the Ethereum ecosystem.

Algorithm 2 Send Validation Completion

```
// Function to mark validation as completed
function sendValidationCompletion(address
walletAddress, string memory cid) external {
// Check if a validation request exists for the given
walletAddress
if (validationRequests[walletAddress].walletAddress !=
address(0)) {
// Check if the validation request has not already been
completed
if (!validationRequests[walletAddress].isvalidated) {
// Mark the validation as completed by updating the
isvalidated flag
validationRequests[walletAddress].isvalidated =
true;
// Emit ValidationCompleted event
emit ValidationCompleted(walletAddress, cid);
} else {
// Error: Validation request already completed
revert("Validation request already completed");
}
} else {
// Error: No validation request found
revert("No validation request found");
}
}
```

The logic for the "Send Validation Completion," and the subsequent features would work similarly to how Sarah and John can materialize their transaction in (1) Send validation request.

(3) Get validation status

With the "Get Validation Status" feature, users can easily inquire about the current status of a validation process or transaction within the system. This functionality provides real-time updates on the progress of actions awaiting validation or confirmation, offering users insights into the state of their transactions. By having access to up-to-date information, users can make informed decisions and respond proactively to any pending validation tasks.

(4) Get CID

The "Get CID" function enables users to obtain the Content Identifier (CID) associated with specific data or digital assets stored on the Ethereum blockchain. CIDs serve as unique identifiers, ensuring secure and tamper-proof referencing of data. With this functionality, users can easily access and verify the integrity of data stored on the blockchain, promoting transparency and trust within the system.

Algorithm 3 Get Validation Status

```
// Algorithm to get the CID associated with a walletAddress
Function getCID(address walletAddress) external view
returns (string memory) {
    // Retrieve the ValidationRequest for the given
    // walletAddress
    ValidationRequest memory request = validationRequests[walletAddress];
    // Check if the validation is completed (isValidated is
    true)
    if (request.isValidated) {
        // Return the CID (Content Identifier) associated with
        // the walletAddress
        return request.cid;
    } else {
        // Throw an error - Validation not completed
        throw "Validation not completed";
    }
}
```

4.2 Confidentiality and integrity of data

The proposed system incorporates several mechanisms to ensure the confidentiality and integrity of data, as well as to prevent unauthorized access or use of the data. Here's how it achieves these objectives:

Algorithm 4 Get CID

```
// Algorithm to get the CID associated with a walletAddress
function getCID(address walletAddress) external view
returns (string memory) {
    // Retrieve the ValidationRequest for the given
    // walletAddress
    ValidationRequest memory request = validationRequests[walletAddress];
    // Check if the validation is completed (isValidated is
    true)
    if (request.isValidated) {
        // Return the CID (Content Identifier) associated with
        // the walletAddress
        return request.cid;
    } else {
        // Throw an error - Validation not completed
        throw "Validation not completed";
    }
}
```

At its foundation, the system leverages Content Identifiers (CIDs), unique cryptographic hashes derived directly from the data content. Any alteration to the data results in a fundamentally distinct CID, an approach that bolsters security, assuring users of data integrity.

Further, within the Ethereum blockchain ecosystem, the system relies on smart contracts endowed with stringent access controls. These contracts permit interactions only to authorized users with requisite permissions, effectively deterring unauthorized access and fortifying data integrity. This access control framework establishes clear demarcations, safeguarding data confidentiality.

Confidentiality within the system also hinges on the secure management of Ethereum wallets and their associated private keys. These keys, vital for transaction signing and network engagement, are secured within wallets designed with high levels of confidentiality - access is confined to wallet owners,

preventing unauthorized entry and ensuring transactional and contractual security.

The system also works via a thorough validation workflow, requiring users to confirm actions or tasks. Authorization is pivotal, permitting only privileged individuals to initiate confirmations. This exacting access control mechanism ensures that data validation is entrusted solely to authorized personnel, bolstering data security.

4.3 Potential performance issues and constraints

The proposed approach foresees some constraints and potential performance issues when handling data and scaling up to a large number of users or transactions.

4.3.1 Performance issues

While IPFS is suitable for storing a wide range of data types, it is important to note that very large files or datasets may present challenges. IPFS divides files into smaller chunks, and very large files can result in a large number of chunks, which may impact network and storage resources [30].

Further, retrieving data from IPFS can be slower compared to traditional centralized storage solutions. IPFS relies on a distributed network of nodes, and the speed of retrieval may depend on network congestion and the availability of nodes hosting the requested content.

Ethereum has historically faced scalability issues, with limited transaction throughput. As the number of users and transactions increases, congestion and slower confirmation times can occur. Ethereum 2.0, an upgrade to the network, aims to address these issues, but it may still be a concern on the current Ethereum chain.

4.3.2 Gas costs

The execution of smart contracts on Ethereum incurs gas costs, which are paid in Ether. Complex operations or frequent interactions with smart contracts can result in high gas fees. This cost may affect the affordability of using the system, especially during periods of high network activity.

4.3.3 Data privacy

Data stored on IPFS is public by default. While the content is addressed cryptographically, anyone can access and retrieve data if they have the CID [31] - ensuring the privacy and security of sensitive data is a concern.

The approach also assumes that users have a reliable internet connection to interact with both IPFS and Ethereum. IPFS, in particular, should be able to handle intermittent network connectivity, but a completely offline user might face challenges.

5. IMPLEMENTATION

The experiment is implemented using a Latitude 3330 laptop with 4 Intel CPUs and deployed on an Ethereum test net using Remix IDE provided by Ethereum Foundation. The experiment involved two parties, a resident and a Notarial Office, responsible for giving out certificates, specifically birth certificates. The experiment's objective involves examining the process, utility, and practicality of implementing blockchain technology for issuing certificates through the Notarial office.

The smart contract created was deployed on the Sepolia

Testnet [32], and enabled the resident to upload the relevant documents needed for validation by the Notarial Office. However, doing this on Ethereum would have been expensive as huge chunks of data cost more Ether in processing fees. Therefore, to be more economical and affordable [33], the resident uploads the document to IPFS to generate a hash and then uploads the hash (CID) to Ethereum. Afterward, the smart contract sends the hash to the Notarial office address, which then decrypts the hash on IPFS to verify identity. After verification, the Notarial Office then sends the birth certificate hash back to the resident's address.

6. RESULTS AND DISCUSSION

The experiment conducted involved two entities: a resident and a Notarial Office responsible for issuing certificates, primarily birth certificates. The primary objective was to assess how blockchain technology can integrate into the certificate issuance process conducted by the Notarial Office. First, a smart contract was developed and set up on the Ethereum Sepolia testnet. This contract empowered the resident to submit the necessary documents for validation by the Notarial Office. Given that huge blocks of data incur higher Ether in fees on the blockchain, the resident optimizes cost-effectiveness by uploading the document to IPFS, generating a hash, and subsequently transmitting this hash (CID) to the blockchain via the smart contract. The smart contract then forwards the hash to the address of the Notarial Office. Subsequently, the Notarial Office decrypts the hash located on IPFS to authenticate it. Once validated, the Notarial Office dispatches the birth certificate to the resident's address.

While experimenting, four key aspects were meticulously documented: Scalability, Encryption and Decryption, Deployment, and the Execution time of the contract. The overall performance of the blockchain was also scrutinized.

6.1 Scalability

For scalability, in Figure 4 we assumed the resident uploaded four documents, CIDs, his personal information, his parents' marriage certificates, and the paper of his birth from the hospital:

Scalability = TPS against Resident & Notarial office.

A transaction from the resident takes 1 sec, and it's 0.8 secs from the Notarial Office.

As the resident made four transactions for each document, the resident's TPS= 3 secs and N.O = 0.8 since the N.O transacts only once after verification.

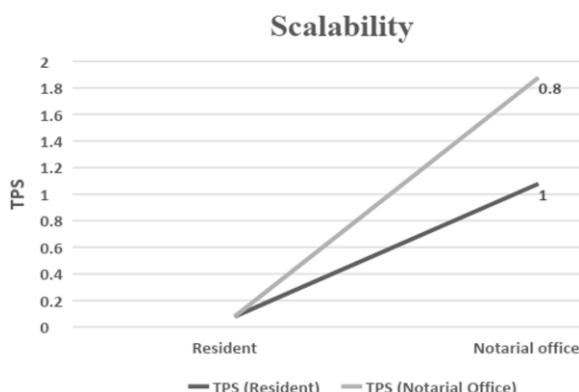


Figure 4. Scalability

6.2 Encryption and decryption results

6.2.1 Using IPFS

In Figure 5, these were experimented off-chain. They occur inside IPFS. Assuming the resident uploaded the four documents and their size is 4 MB in total as experimented, encryption time is 5 secs and decryption time is 7 secs.

Table 1 presents performance data for encryption and decryption processes concerning different content sizes. The content sizes range from 250 KB to 4 MB. The time taken for encryption and decryption is measured in seconds. For content sizes of 250 KB and 500 KB, both encryption and decryption operations take approximately 1 second each. As the content size increases to 1 MB and 2 MB, the encryption and decryption times also slightly increase, taking around 2 seconds and 3 seconds, respectively.

For the largest content size of 4 MB, encryption requires approximately 5 seconds, while decryption takes about 7 seconds. The data indicates that, in general, larger content sizes tend to incur slightly longer encryption and decryption times. However, the differences between the times for different content sizes are relatively small in this dataset. It is essential to consider the encryption algorithm and computational capabilities when evaluating the actual performance of encryption and decryption operations.

Table 1. Document size

Content (Size)	Encryption (secs)	Decryption (secs)
250 KB	1	1
500KB	1	1
1 MB	2	2.3
2 MB	3	3.4
4 MB	5	7

Encryption and Decryption Performance Using IPFS

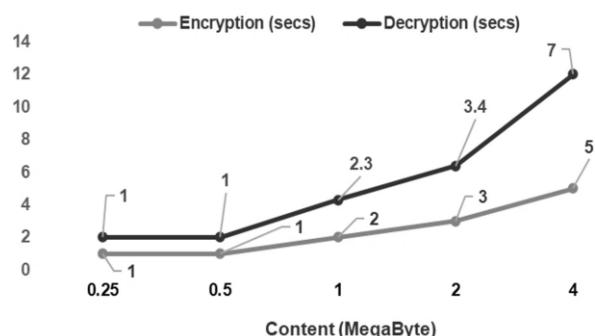


Figure 5. Encryption and decryption performance using IPFS

6.2.2 Using StorJ

The StorJ architecture is so flexible I can't see us ever going back to anything else. For any type of serious application, it just makes sense to build it on Storj. Its flexibility lets us easily work with or move to different platforms, or redirect traffic in different ways.

Table 2 and Figure 6 show that as document sizes vary, encryption times are influenced by the document's magnitude, increasing with larger files. If you see the Figure 6. In contrast, decryption times exhibit less sensitivity to document size, remaining relatively constant. This data has implications for systems where document security and processing speed are vital factors, as it provides insights into the time requirements for encrypting and decrypting documents of varying sizes.

Table 2. Document size

Content (Size)	Encryption (secs)	Decryption (secs)
250 KB	6	1
500KB	6	1.2
1 MB	7	1.2
2 MB	13	1.3
4 MB	22	1.3

Encryption and Decryption Performace Using StorJ

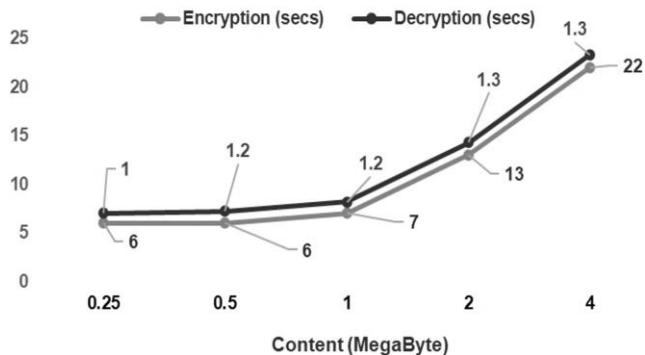


Figure 6. Encryption and decryption performance using StorJ

6.2.3 Assessing encryption and decryption performance: IPFS vs. StorJ

IPFS is more ideal because it has faster encryption and decryption time than StorJ. Moreover, IPFS secures files better than StorJ. The recommended off-chain storage system, according to the experiment, is IPFS.

Table 3. Results from IPFS and StorJ

Content (Size)	Encryption (secs)	Decryption (secs)
250 KB	1	1
500KB	1	1
1 MB	2	2.3
2 MB	3	3.4
4 MB	5	7

IPFS Result		
Content (Size)	Encryption (secs)	Decryption (secs)
250 KB	1	1
500KB	1	1
1 MB	2	2.3
2 MB	3	3.4
4 MB	5	7

StorJ Result		
Content (Size)	Encryption (secs)	Decryption (secs)
250 KB	1	1
500KB	1	1
1 MB	2	2.3
2 MB	3	3.4
4 MB	5	7

Companson Between IPFS and StorJ

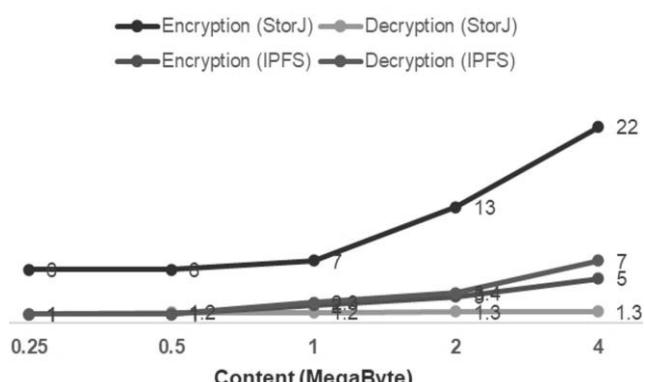


Figure 7. Comparison between IPFS and StorJ

In Table 3 and Figure 7, we evaluated the encryption and decryption performance of IPFS and StorJ for various document sizes. IPFS consistently showed faster results, with encryption and decryption times ranging from 1 to 5 seconds. In contrast, StorJ had slightly longer encryption and decryption times, ranging from 1 to 7 seconds for the same document sizes. Overall, IPFS outperformed StorJ in terms of speed for these operations.

6.2.4 Deployment & execution time

It means when we deploy a contract on Remix IDE, the time taken for deployment to be confirmed by the blockchain is 10 secs at first, then the second time of deployment confirmation is 5, and so on. In Figure 8, the deployment of the smart contract and the execution of its function were calculated. The average deployment time is 9 secs, while the average execution time is 6 secs. The deployment time and execution time were performed 6 times to get the average calculations. And that's how it's been calculated. If I were to continue the experiment. Their average will continue to be 9 and 6.

■ Deploy. Time(secs) ■ Exec. Time(secs)

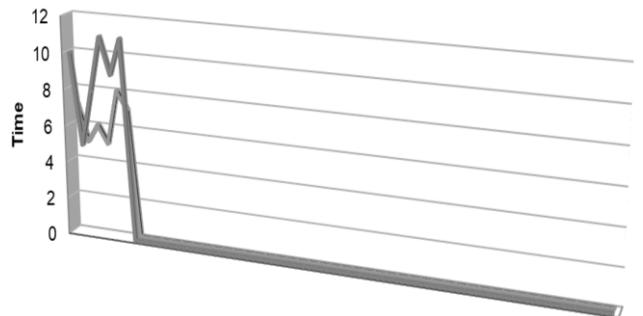


Figure 8. Deployment & execution time using remix IDE

7. MULTI REQUEST MULTI-LOCATION

In Figure 9, within the context of distributed notarial services, where diverse geographical locations house numerous notarial offices, the imperative of judiciously routing incoming requests underscores operational efficiency. This exposition introduces a pioneering location-based routing mechanism, intricately woven through the amalgamation of Remix IDE and MetaMask. This synergistic fusion orchestrates the optimization of routing dynamics, meticulously entwining proximity, capacity, and power supply considerations, thereby orchestrating a seamless and calibrated allocation of requests across multifarious notarial domains. The architecture of this system is tripartite, constituted by:

-Remix IDE: An Ethereum smart contract-focused integrated development environment, serving as the genesis of the routing smart contract's inception and deployment.

-MetaMask: A cryptocurrency wallet and gateway, adeptly bridging user interface interactions with the smart contract realm, enabling coherent exchanges.

-Smart Contract: Ingeniously synthesized via Remix IDE, this repository encapsulates the intricate logic underpinning request routing predicated upon predefined evaluative benchmarks.

7.1 Workflow, benefits, and conclusion

In Figure 9, the procedural trajectory of this location-based routing system is elegantly choreographed:

1. Users initiate the sequence by submitting requests via the dedicated user interface.

2. The ensuing orchestration unfolds as the smart contract meticulously assesses notarial office locations, assimilating variables like proximity, office capacity, and power supply status.

3. Guided by this comprehensive assessment, the smart contract tactfully ascertains the optimally aligned notarial office for each incoming request, thus engendering the pragmatic allocation of resources.

4. MetaMask seamlessly interfaces between the user interface and the smart contract, certifying the assignment of designated offices for individual requests.

This innovative paradigm proffers several discernible

benefits:

-Efficient Distribution: Automated routing minimizes latency, effectively allocating TPS requests to suitable notarial offices and judiciously managing resource utilization.

-Proximity Optimization: Nearest available office service users, enhancing service quality by minimizing travel time.

-Resource Balance: By assessing capacity and power supply, an equitable distribution of office workload is maintained, ensuring optimal resource utilization. In summation, the evoked location-based routing system, borne from the collaborative prowess of Remix IDE and MetaMask, serves as a comprehensive solution for the optimal routing of requests across varied notarial domains. Its emphasis on proximity, capacity, and power supply harmoniously enriches user experience while attaining a felicitous equilibrium in resource allocation. This discourse imparts a synthesized understanding of the system's architecture, its functional sequence, and the accrued merits intrinsic to this innovative routing paradigm.

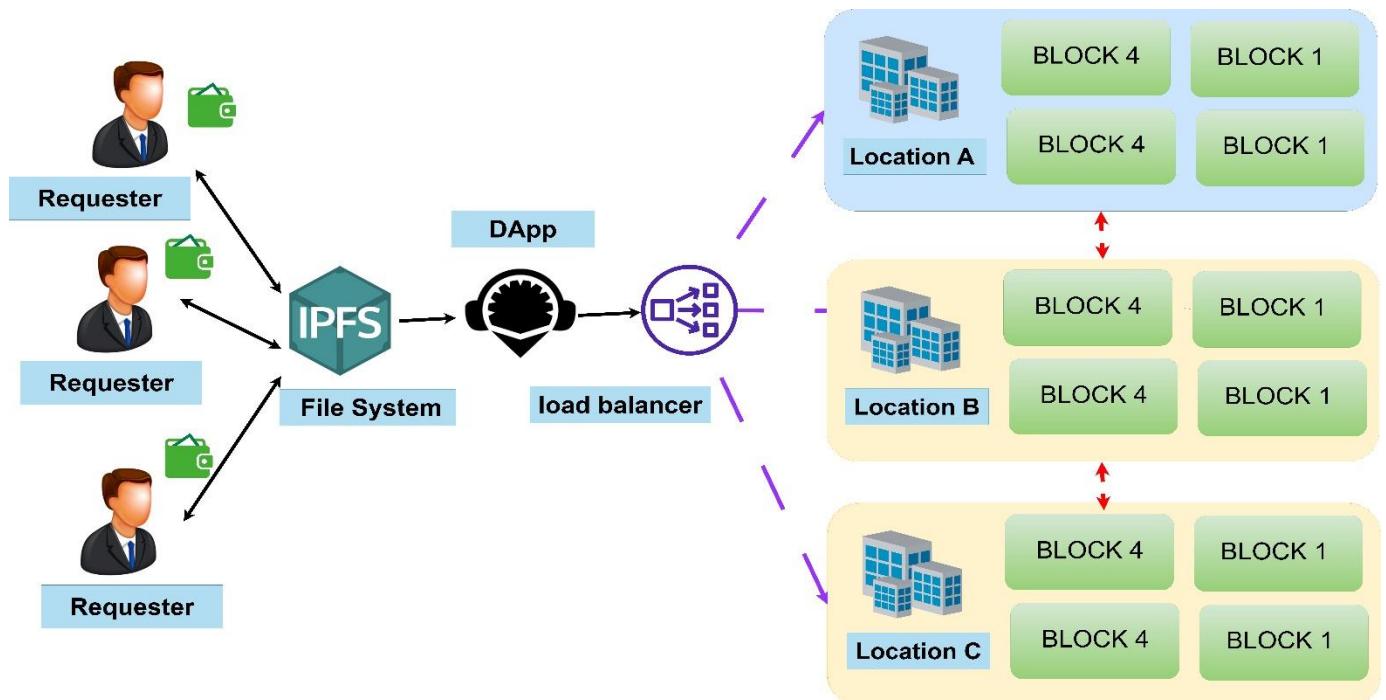


Figure 9. Multi-request multi-location system high-level architecture

7.2 Performance transactions/block confirmation

TPS (transactions per second) is calculated by number of transactions/Block confirmation time. For example: 5 transactions were confirmed by the block in 2, 3, 4, 5 secs. Factors that affect Block confirmation time: Network Congestion, Block processing time, Speed of the PC used, etc. The equation for calculating TPS (transactions per second) when considering N transactions can be expressed as:

$$TPS = \frac{N}{\text{Block Confirmation Time}}$$

where,

N is the total number of transactions confirmed in the block.

Block Confirmation Time is the time taken for the block to be confirmed, which is influenced by factors like network congestion, block processing time, and the speed of the PC used.

Table 4. Performance transactions/Block confirmation

Number of Users/Requests	Block Confirmation
2	2
3	3
5	4
5	5
	3

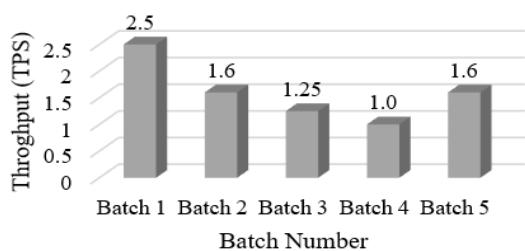


Figure 10. Performance transactions/block confirmation

In Table 4, and Figure 10, we measured the time taken for block confirmation for different numbers of users/requests. With 5 users/requests, block confirmations ranged from 2 to 5 seconds. This data demonstrates the system's ability to handle multiple transactions with varying confirmation times.

Optimizing TPS has several potential benefits for the system and user experience:

-A higher TPS allows the system to handle a larger number of transactions concurrently. In a notarial services system that serves diverse geographical locations with numerous notarial offices, scalability is essential [34]. As the number of users and requests increases, a higher TPS ensures that the system can efficiently process and route those requests without significant delays. This scalability is essential for accommodating growth and fluctuations in demand.

-A higher TPS equally translates to faster transaction processing. Users submitting requests to the system will experience quicker response times and reduced waiting periods. In the context of notarial services, where timely document verification and authentication are critical, a faster response time enhances the overall user experience. Users are more likely to be satisfied with a system that provides rapid service, improving customer satisfaction and trust in the service.

-In a distributed system with multiple notarial offices across different locations, minimizing latency is vital. A higher TPS helps reduce the time it takes for a request to be processed and routed to the appropriate notarial office. Reduced latency ensures that users receive services promptly, and it's particularly important when considering the proximity optimization aspect of the system. Users are more likely to be directed to nearby notarial offices promptly.

8. CONCLUSION

In the conducted research, we embarked on an experiment involving a resident and a Notarial Office to explore the potential integration of blockchain technology into the certificate issuance process. The focal point of the experiment was the development of a smart contract on the Ethereum Sepolia testnet, which enabled residents to securely submit documents for validation by the Notarial Office. Employing the IPFS for document upload and hash transmission, the resident optimized for cost-efficiency. The experiment documented crucial facets including Scalability, Encryption and Decryption, Deployment, and Execution time of the contract. Notably, IPFS emerged as the favored off-chain storage mechanism due to its superior encryption and decryption performance compared to alternatives like StorJ. Furthermore, the research emphasized the significance of considering factors such as transaction processing rates, deployment and execution times, and the overall blockchain performance, influenced by elements like network congestion and hardware capabilities. This comprehensive investigation sheds light on the viable incorporation of blockchain technology to improve government service administration - certificate issuance in this case - offering valuable insights for similar endeavors.

9. FUTURE WORK

An essential avenue for future research and development

that stemmed from our experimental study on optimizing government administrative efficiency through Ethereum smart contracts and the Interplanetary File System (IPFS) is the investigation of a hybrid architecture. This approach holds the potential to address data security concerns raised by government institutions while preserving the benefits of blockchain technology.

9.1 Exploring a hybrid architecture for data security

Recognizing the data security and sovereignty concerns of governments, a prime future direction could involve the design, implementation, and evaluation of a hybrid architecture that integrates both permissionless (public) and permissioned (private) blockchain networks [35]. By incorporating permissioned components into the solution, governments can exert greater control over participant access and data visibility. This approach strikes a balance between the decentralization advantages of public blockchains and the data security assurances of private networks.

In this setup, the administrative process could leverage the efficiency of Ethereum smart contracts while operating within a permissioned environment. This controlled ecosystem would allow governments to manage access rights, monitor data flows, and ensure compliance with regulatory requirements. Additionally, the hybrid architecture could facilitate secure integration with existing government systems, streamlining the transition to decentralized administrative processes without disrupting established workflows.

In our future research, we will focus on several key areas. Firstly, crafting a hybrid blueprint detailing the interaction between permissionless and permissioned components alongside the protocols relevant for seamless data exchange. This approach would also need devising mechanisms outlining roles, responsibilities, and data access levels for the permissioned blockchain, in addition to leveraging robust access controls to counter unauthorized data exposure.

9.2 Interoperability with existing systems

For the envisioned integration between the blockchain-based administrative solution and legacy government databases, there will be a need to overcome compatibility challenges, towards interoperability. To achieve this, a comprehensive design and implementation of standardized Application Programming Interfaces (APIs) and middleware layers could be used [36]. These components will act as bridges between the proposed blockchain-based administrative solution and the legacy government databases, enabling smooth and secure data exchange.

This approach would involve designing standardized APIs to establish a well-defined interface for communication between the blockchain solution and legacy systems. These APIs should accommodate various data types, including personal information, certificates, and transaction records. Additionally, middleware layers could be strategically positioned to manage data translation, protocol conversion, and communication optimization. This architecture ensures that data transmitted between different systems remains consistent, regardless of the platforms' underlying structures.

9.3 Potential challenges and remedies

The research approach's potential limitations and challenges

include scalability and performance concerns. Government administrative processes frequently handle substantial data volumes and transactions, posing challenges for blockchain networks. These challenges could be slow transaction processing times or elevated fees during network congestion.

Addressing these issues necessitates efficient consensus mechanisms like proof-of-stake and optimizing smart contract code. Layer-2 scaling solutions, such as sidechains and state channels, can alleviate scalability challenges, ensuring responsiveness under heavy workloads [37]. Data privacy and compliance are crucial when managing sensitive government data on a transparent blockchain, while robust encryption methods and access controls come in handy in safeguarding sensitive data.

For security concerns that may arise in hybrid architectures combining public and private blockchains, stringent access controls, end-to-end encryption, and regular security audits could be employed to mitigate these risks.

REFERENCES

- [1] Mohamed, A.A.D., Alkhateeb, Y.M., Agarwal, P., Abdelwahab, A.R. (2022). Characteristics of blockchain and smart services, for smart governments: A systematic review of the literature. *International Journal of Information Systems and Project Management*, 10(3): 30-55. [10.12821/ijispdm100302](https://doi.org/10.12821/ijispdm100302)
- [2] Wagola, R., Nurmandi, A., Misran, Subekti, D. (2023). Government digital transformation in Indonesia. In 25th International Conference on Human-Computer Interaction, HCII 2023, Copenhagen, Denmark, pp. 286-296. https://doi.org/10.1007/978-3-031-36001-5_37
- [3] Othman, M.H., Razali, R., Nasrudin, M.F. (2020). Key factors for e-government towards sustainable development goals. *International Journal of Advanced Science and Technology*, 29(6): 2864-2876
- [4] Guo, Y. G., Yin, Q., Wang, Y., Xu, J., Zhu, L. (2023). Efficiency and optimization of government service resource allocation in a cloud computing environment. *Journal of Cloud Computing*, 12(1): 18. <https://doi.org/10.1186/s13677-023-00400-2>
- [5] Shrier, D., Wu, W., Pentland, A. (2016). Blockchain & infrastructure (identity, data security). *Massachusetts Institute of Technology-Connection Science*, 1(3): 1-19.
- [6] Lykidis, I., Drosatos, G., Rantos, K. (2021). The Use of Blockchain Technology in e-Government Services. *Computers* 2021, 10(12), 168. <https://doi.org/10.3390/COMPUTERS10120168>
- [7] Abdelhamid, I.R., Halim, I.T.A., Ali, A.E.M.A., Ibrahim, I.A. (2023). A survey on blockchain for intelligent governmental applications. *Indonesian Journal of Electrical Engineering and Computer Science*, 31(1): 501-513. <https://doi.org/10.11591/IJEECS.V31.I1.PP501-513>
- [8] Hou, H. (2017). The application of blockchain technology in E-government in China. In 2017 26th International Conference on Computer Communication and Networks (ICCCN), Vancouver, BC, Canada, pp. 1-4. <https://doi.org/10.1109/ICCCN.2017.8038519>
- [9] Khan, S., Shaal, M., Majdalawieh, M., Nizamuddin, N., Nicho, M. (2022). Blockchain for governments: The case of the Dubai government. *Sustainability*, 14(11): 6576. <https://doi.org/10.3390/SU14116576>
- [10] Pinter, K., Schmelz, D., Lamber, R., Strobl, S., Grechenig, T. (2019). Towards a multi-party, blockchain-based identity verification solution to implement clear name laws for online media platforms. *Lecture Notes in Business Information Processing*, 361: 151-165. https://doi.org/10.1007/978-3-030-30429-4_11
- [11] Parra Moyano, J., Ross, O. (2017). KYC optimization using distributed ledger technology. *Business & Information Systems Engineering*, 59: 411-423. <https://doi.org/10.1007/s12599-017-0504-2>
- [12] Negara, E.S., Hidayanto, A.N., Andryani, R., Erlansyah, D. (2021). A survey blockchain and smart contract technology in government agencies. *IOP Conference Series: Materials Science and Engineering*, 1071(1): 012026. <https://doi.org/10.1088/1757-899X/1071/1/012026>
- [13] Insight Report. (2020). Exploring blockchain technology for government transparency: Blockchain-based public procurement to reduce corruption. https://www3.weforum.org/docs/WEF_Blockchain_Government_Transparency_Report.pdf.
- [14] Habib, G., Sharma, S., Ibrahim, S., Ahmad, I., Qureshi, S., Ishfaq, M. (2022). Blockchain technology: Benefits, challenges, applications, and integration of blockchain technology with cloud computing. *Future Internet*, 14(11): 341. <https://doi.org/10.3390/FI14110341>
- [15] Edward, S. (2019). The policy environment for blockchain innovation and adoption. <https://policycommons.net/artifacts/3808668/the-policy-environment-for-blockchain-innovation-and-adoption/4614615/>.
- [16] Khatal, S., Rane, J., Patel, D., Patel, P., Busnel, Y. (2021). Fileshare: A blockchain and IPFS framework for secure file sharing and data provenance. In *Advances in Machine Learning and Computational Intelligence: Proceedings of ICMLCI 2019*, pp. 825-833. https://doi.org/10.1007/978-981-15-5243-4_79
- [17] Vergne, J.P. (2020). Decentralized vs. distributed organization: Blockchain, machine learning and the future of the digital platform. *Organization Theory*, 1(4): 2631787720977052. <https://doi.org/10.1177/2631787720977052>
- [18] Mattila, J. (2016). The blockchain phenomenon - The disruptive potential of distributed consensus architectures. <https://www.econstor.eu/handle/10419/201253>
- [19] Atzori, M. (2017). Blockchain technology and decentralized governance: Is the state still necessary? *Journal of Governance and Regulation*, 6(1): 45-62. https://doi.org/10.22495/jgr_v6_i1_p5
- [20] Solidity - Solidity v0.8.21 documentation. <https://docs.soliditylang.org/en/v0.8.21/>.
- [21] Benet, J. (2014). IPFS-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561*. <https://doi.org/10.48550/arXiv.1407.3561>
- [22] Storj: A decentralized cloud storage network framework. <https://github.com/storj/whitepaper>
- [23] Welcome to Remix's documentation! - Remix - Ethereum IDE 1 documentation. <https://remix-ide.readthedocs.io/en/latest/>.
- [24] Hassanein, A.A., El-Tazi, N., Mohy, N.N. (2022). Blockchain, smart contracts, and decentralized applications: An introduction. *Implementing and Leveraging Blockchain Programming*, pp. 97-114.

- https://doi.org/10.1007/978-981-16-3412-3_6
- [25] Cong, L.W., He, Z. (2019). Blockchain disruption and smart contracts. *The Review of Financial Studies*, 32(5): 1754-1797. <https://doi.org/10.2139/SSRN.2985764>.
- [26] Gas and fees. <https://ethereum.org/en/developers/docs/gas/>.
- [27] Rouhani, S., Deters, R. (2017). Performance analysis of ethereum transactions in private blockchain. In 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, pp. 70-74. <https://doi.org/10.1109/ICSESS.2017.8342866>
- [28] Wu, K., Ma, Y., Huang, G., Liu, X. (2021). A first look at blockchain-based decentralized applications. *Software: Practice and Experience*, 51(10): 2033-2050. <https://doi.org/10.1002/spe.2751>
- [29] Doan, T.V., Psaras, Y., Ott, J., Bajpai, V. (2022). Towards decentralised cloud storage with IPFS: Opportunities, challenges, and future directions. arXiv preprint arXiv:2202.06315. <https://doi.org/10.48550/arXiv.2202.06315>
- [30] Fakultät, M.N. (2023). Analysis and comparison of deduplication strategies in IPFS. Master's thesis, Humboldt University of Berlin.
- [31] IPFS - Privacy and encryption. <https://docs.ipfs.tech/concepts/privacy-and-encryption/#encryption>.
- [32] Networks. <https://ethereum.org/nb/developers/docs/networks/>.
- [33] Ismail, A., Toohey, M., Lee, Y.C., Dong, Z., Zomaya, A. Y. (2022). Cost and performance analysis on decentralized file systems for blockchain-based applications: State-of-the-art report. In 2022 IEEE International Conference on Blockchain (Blockchain), Espoo, Finland, pp. 230-237. <https://doi.org/10.1109/Blockchain55522.2022.00039>
- [34] Liu, F., He, S., Li, Z., Xiang, P., Qi, J., Li, Z. (2023). An overview of blockchain efficient interaction technologies. *Frontiers in Blockchain*, 6: 996070. <https://doi.org/10.3389/fbloc.2023.996070>
- [35] Ghosh, B.C., Bhartia, T., Addya, S.K., Chakraborty, S. (2021). Leveraging public-private blockchain interoperability for closed consortium interfacing. In IEEE INFOCOM 2021 - IEEE Conference on Computer Communications, Vancouver, BC, Canada, pp. 1-10. <https://doi.org/10.1109/INFOCOM42981.2021.9488683>
- [36] Bokolo, A.J. (2022). Exploring interoperability of distributed Ledger and Decentralized Technology adoption in virtual enterprises. *Information Systems and e-Business Management*, 20(4): 685-718. <https://doi.org/10.1007/s10257-022-00561-8>
- [37] Bottoni, S., Datta, A., Franzoni, F., et al. (2023). 1DLT: Rapid deployment of secure and efficient EVM-based blockchains. In 4th International Conference on Blockchain Economics, Security and Protocols (Tokenomics 2022), pp. 1-15. <https://doi.org/10.4230/OASIcs.Tokenomics.2022.3>