Exploration of Drug Science

Open Exploration

Open Access Systematic Review

Utilizing the Ethereum blockchain for retrieving and archiving augmented reality surgical navigation data

Sai Batchu[1], Michael J. Diaz[2], Lauren Ladehoff[3], Kevin Root[2], Brandon Lucke-Wold[4]*

[1]Cooper Medical School, Rowan University, Camden, NJ 08103, USA
[2]College of Medicine, University of Florida, Gainesville, FL 32611, United States
[3]Morsani College of Medicine, University of South Florida, Tampa, FL 33620, United States
[4]Department of Neurosurgery, University of Florida, Gainesville, FL 32611, United States

***Correspondence:** Brandon Lucke-Wold, Department of Neurosurgery, University of Florida, Gainesville, FL 32611, United States. Brandon.Lucke-Wold@neurosurgery.ufl.edu

Academic Editor: Francisco Javier Luque Garriga, University of Barcelona, Spain

Received: November 8, 2022 **Accepted:** January 17, 2023 **Published:** February 28, 2023

Cite this article: Batchu S, Diaz MJ, Ladehoff L, Root K, Lucke-Wold B. Utilizing the Ethereum blockchain for retrieving and archiving augmented reality surgical navigation data. Explor Drug Sci. 2023;1:55–63. https://doi.org/10.37349/eds.2023.00005

Abstract

Aim: Conventional techniques to share and archive spinal imaging data raise issues with trust and security, with novel approaches being more greatly considered. Ethereum smart contracts present one such novel approach. Ethereum is an open-source platform that allows for the use of smart contracts. Smart contracts are packages of code that are self-executing and reside in the Ethereum state, defining conditions for programmed transactions. Though powerful, limited attempts have been made to showcase the clinical utility of such technologies, especially in the pre- and post-operative imaging arenas. Herein, we therefore aim to propose a proof-of-concept smart contract that stores intraoperative three-dimensional (3D) augmented reality surgical navigation (ARSN) data and was tested on a private, proof-of-authority network. To the author's best knowledge, the present study represents a first-use case of the InterPlanetary File Storage protocol for storing and retrieving spine imaging smart contracts.

Methods: The content identifier hashes were stored inside the smart contracts while the interplanetary file system (IPFS) was used to efficiently store the image files. Insertion was achieved with four storage mappings, one for each of the following: fictitious patient data, specific diagnosis, patient identity document (ID), and Gertzbein grade. Inserted patient observations were then queried with wildcards. Insertion and retrieval times for different record volumes were collected.

Results: It took 276 milliseconds to insert 50 records and 713 milliseconds to insert 350 records. Inserting 50 records required 934 Megabyte (MB) of memory per insertion with patient data and imaging, while inserting 350 records required almost the same amount of memory per insertion. In a database of 350 records, the retrieval function needs about 1,026 MB to query a record with all three fields left blank, but only 970 MB to obtain the same observation from a database of 50 records.

Conclusions: The concept presented in this study exemplifies the clinical utility of smart contracts and off-chain data storage for efficient retrieval/insertion of ARSN data.

Keywords

Spine, blockchain, Ethereum, smart contracts, Solidity, interplanetary file system, spinal imaging

Introduction

Patient imaging records are increasingly being stored piecemeal with different parties as modern information systems in healthcare trend towards control by multiple entities. This reliance on multiple parties to host and share protected data, especially in emerging fields, including computer-assisted spinal surgery, increases the opportunity for security [1–3] and transparency [4–7] issues. Further, due to the concern of time-sensitive data not being available when necessary, healthcare data is often stored in permission databases with multiple levels. The concerns warrant the exploration of novel solutions, such as the utilization of blockchain technology.

A blockchain is a distributed, decentralized ledger similar to a digital append-only list constituting ordered sequences of grouped entries, named "blocks". These blocks are ordered in a sequence ordered through the storage of a cryptographic hash of the proceeding block in the following one (Figure 1). Notably, the deterministic and irreversible nature of these hash functions permits the blocks of data to be "chained" together. Further, since the alteration of any record alters the hash of the current block in addition to any subsequent blocks thereafter, this paradigm ensures data stored within the system is immutable. The blockchain is shared amongst "nodes", or computers without previous relationships linked via peer-to-peer networks. Significantly, no individual node can control the data. The nature of this distributed architecture permits decentralization and precludes a central point of failure. Moreover, the blockchain is fully transparent with each transaction being transmitted to all nodes and users, who can verify the transactions.

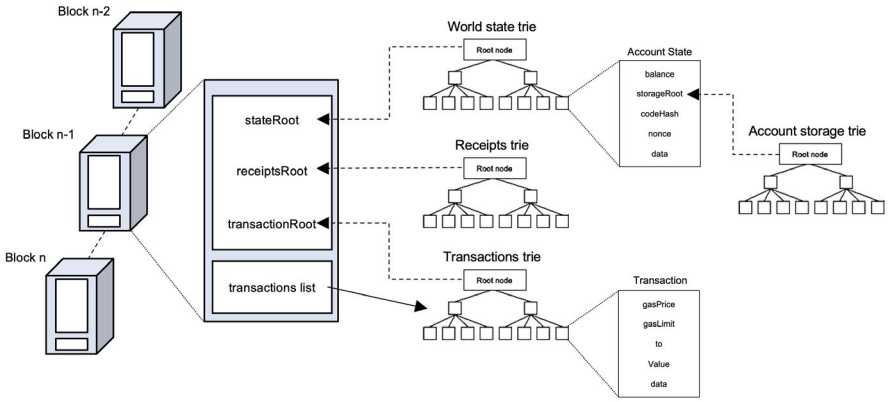

Figure 1. Detailed overview of the Ethereum blockchain

Note. Adapted from "Using ethereum smart contracts to store and share COVID-19 patient data," by Batchu S, Patel K, Henry OS, Mohamed A, Agarwal AA, Hundal H, et al. Cureus. 2022;14:e21378 (https://www.cureus.com/articles/72568-using-ethereum-smart-contracts-to-store-and-share-covid-19-patient-data#!/). CC BY.

Theoretically, blockchain architecture serves as an immutable ledger for any form of data, even medical records. Currently, blockchain is most thought of in the context of Bitcoin, which uses blockchain to record transactions of digital currency [8]. Following this, blockchain technology has become increasingly popular and has since been utilized with the new digital currency, Ethereum. Ethereum is distinct from other blockchains in that it combines the benefits of blockchain technology with software applications as open-source, international. This quality permits developers to generate programs that execute only when specific conditions are met, a concept known as smart contracts [9]. The Ethereum platform can be applied to additional use cases requiring strong security and flexibility, such as preserving medical records, as it permits code to be performed with specific logic and offers data storage not limited to Bitcoin transactions. Although existing illustrations of the potential of the Ethereum blockchain integrating medical data are limited, the early findings are encouraging. One demonstration of this is Gürsoy et al. [10] use of the Ethereum

blockchain to handle pharmacogenomics data composed of gene-variant-drug combinations. However, no research has yet investigated the use of the Ethereum blockchain to store and query imaging data.

Currently, computer-assisted spine surgery data is transported among healthcare professionals via a physical device [e.g., compact disc (CD), digital video disk (DVD), universal serial bus (USB)]. Image Share Network (ISN) was developed as an alternative to cumbersome physical transport. The ISN stores medical photos in a third-party clearinghouse searchable with a hash of a secret token. Within an established period, the token disclosed by the patient can be used to reproduce the unique hash and download the data [11]. ISN is just one of a few existing cloud-based commercial systems in current use [12–14]. While these platforms circumvent physical media transfer, their designs create concerns over the participation of third parties and the centralization of the framework.

In the field of spine surgery, the emerging use of intraoperative three-dimensional (3D) imagery with a navigation system has been introduced to lower the rate of surgical screw malposition. Further, 3D augmented reality surgical navigation (ARSN) has emerged as a way to utilize optical video cameras displaying 3D intraoperative images and a navigation path for screws. However, this novel technology requires significant collaboration between healthcare professionals as well as their secure and immaculate handling of data which can be facilitated by the Ethereum blockchain platform.

Therefore, this study illustrates the viability of a smart contract to incorporate ARSN data into the Ethereum blockchain platform, thereby enhancing patient privacy, data security, and record immutability. The price of storage is a disadvantage of keeping image files on Ethereum [9, 15]. To avoid this, the peer-to-peer distributed interplanetary file system (IPFS) was employed to save the imaging outside of the chain. Concurrently, the associated has was stored on Ethereum. Consequently, IPFS was used to maintain the data image while Ethereum housed the unchangeable history of hash references 3D cone-beam computed tomography (CT) scans from a robot-assisted, guided pedicle screw positioning system were utilized to provide proof of concept. Nonetheless, the given framework is adaptable to any form of spine surgery data.

Materials and methods

Methods

This article provides a detailed use case of memory and speed-effective Ethereum smart contract for storing and accessing image data. The IPFS content-addressing protocol was employed to store the image files, while the IPFS hash was kept as a bytes32 type in the blockchain. All data points were stored as individual transactions on the smart contract. Additionally, the authors describe a broad view of Ethereum and IPFS and additional technical details can be found in the Ethereum yellow paper [15] and the IPFS white paper [16].

Description of Ethereum blockchain and smart contracts

The Ethereum platform is based on modified Merkle Patricia Tries, where the blocks save only the hash of the root node of each trie instead of all the data, while still preserving the immutability of the log (Figure 1). There are four main trie types in Ethereum: the global state, the account storage, the transaction, and the transaction receipt [15]. The global state (world state) trie records transactions and allows the mapping of connections between addresses (accounts) and account states. The account storage trie holds account-specific data and is the repository for smart contract data.

Ethereum is a programmable blockchain due to the presence of smart contracts, which are Turing-complete, self-executing programs which use the Ethereum virtual machine (EVM) and autonomous storage [15]. Contract operations access three types of data storage: stack, memory, and permanent storage. Contrary to stack and memory, the contract's storage continues beyond the conclusion of computation [9]. Every contract call, storage, and retrieval leads to a cost or "gas", paid in Ethereum's native cryptocurrency ether [9]. Solidity, an object-oriented programming language built for EVM [17] is used to write smart contracts.

Consensus techniques enable all network nodes to concur on the transactions that will be put into the blockchain. Proof of work (PoW) consensus needs nodes to "mine" blocks by solving a mathematical challenge utilizing computational energy. The block is added to the chain once it has been solved and

validated by other nodes, and also rewards the node which mined the block [8]. Proof of authority (PoA) consensus, on the other hand, relies on a small group of trustworthy nodes known as "authorities" that are officially permitted to verify transactions and attach new blocks. Such a consensus process offers a more rapid alternative to PoW and is often utilized in private blockchain networks [18].

Description of IPFS

The peer-to-peer, decentralized IPFS technology is used to store files. IPFS uses content addressing, which refers to items (such as photos, documents, and videos) by their hashes on file rather than by the server on which it is stored. Since the hashes are generated cryptographically from the content, the IPFS file address can serve as a substitute for the content [16]. In comparison to conventional hypertext transfer protocol, the hash is utilized to retrieve the appropriate object from the appropriate nodes on the peer-to-peer network. Objects in IPFS are identified by their Base58 encoded hash. IPFS hashes are multihashes, which are hashes that include the hash length and the hash function in the first two bytes of the hash. Merkle directed acyclic graphs (DAGs) are used to structure IPFS objects, making the file system both decentralized and immutable [16].

Network configuration

In order to enable more secure testing and storage away from the public IPFS network, a straightforward private IPFS network with three nodes was set up. Each imaging file that was uploaded to the secure IPFS network contained fictitious patient information. The printed hashes were saved for use in subsequently inserting or querying smart contracts.

The hashes were stored and accessed using smart contracts that were tested in the JavaScript testing environment of the truffle suite v5.1.62, an Ethereum development framework. Since this private network was independent of the main Ethereum public network, testing smart contracts could be done without requiring frequent deployments. The PoA mechanism was used to test the contracts on the six-node blockchain network. The performance of private Ethereum PoA networks is not greatly impacted by different node configurations, hence they were not evaluated [19]. The development's default gas constraints were applied.

Insertion

Iteration would be necessary to obtain observations from data stored in an array. Instead, mappings—which are comparable to hash maps—were used to store the data, enabling faster key-value lookups. Four storage mappings were applied using Solidity's mapping type. Each observation was put into its own struct, a special type that allows the grouping of various variables of various types. Patient information, including the IPFS hash, is included in one mapping. Associated diagnosis (e.g., spondylolisthesis), patient identification, Gertzbein grade, and the other three mappings serve as keys to a variety of counter identity documents (IDs). The counter ID is a global variable that gives each entered observation a distinct index. Consequently, retrieving the patient observation from the database using the specified counter IDs is possible by querying with any of the three keys (Figure 2). For effective querying, this insertion mechanism enables non-iterative lookup.

A single patient record was added using the insertion function for a specific diagnosis, patient ID, and Gertzbein grade (Figure 3). Patient identification, primary diagnosis, screw diameter, Gertzbein grade (reflecting the deviation of the screw from the ideal interpedicular trajectory), and 3D cone-beam CT image identifier hash were all included in the patient record. The insertion function then, if necessary, transforms the patient data to the desired storage type after it has been supplied. The disease-patient identifier-Gertzbein grade combination is added to an array if it does not already exist. The pedicle screw diameter, the patient's unique identity, and the primary diagnosis are then entered into respective mappings. The value array receives the counter ID in addition. The counter ID variable is then changed after inserting the observation struct and key-value pair (made of the counter ID and patient data struct) into the database.

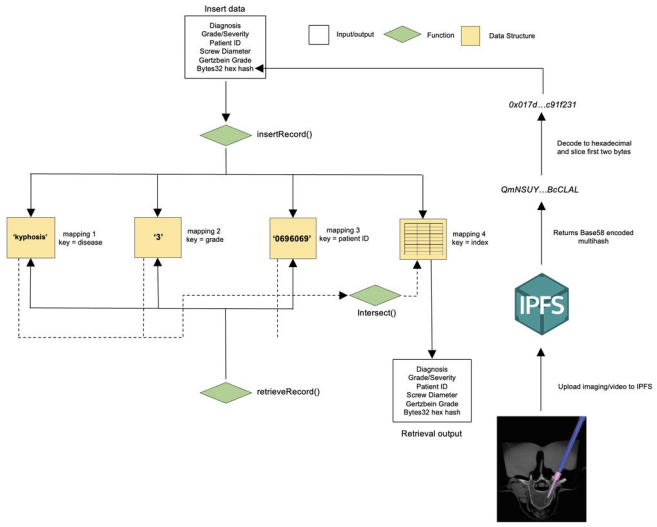

Figure 2. Smart contract architecture. Insertion and retrieval mapping exemplified with mock ARSN input to IPFS

Note. Adapted from "Using ethereum smart contracts to store and share COVID-19 patient data," by Batchu S, Patel K, Henry OS, Mohamed A, Agarwal AA, Hundal H, et al. Cureus. 2022;14:e21378 (https://www.cureus.com/articles/72568-using-ethereum-smart-contracts-to-store-and-share-covid-19-patient-data#!/). CC BY.

Algorithm 1 Pseudocode for inserting observation

```
function INSERTOBSERVATION( STRING diagnosis, UINT grade, UINT patientID, UINT screwDiameter, STRING GertzbeinGrade, bytes32 ipfsHash)
    diagnosisGrade ← diagnosis.uintToBytes32
    patientID ← patientID.uintToBytes32
    screwDiameter ← screwDiameter.uintToBytes32
    GertzbeinGrade ← GertzbeinGrade.stringToBytes32
    if observation exists in database then
        uniqueObservations[i].push(diagnosis, patientID, GertzbeinGrade)
        diagnosisMapping[diagnosis].push(counterID)
        patientIDMapping[patientID].push(counterID)
        GertzbeinGradeMapping[GertzbeinGrade].push(counterID)
        database[counterID].push(counterID)
        counterID++
    end if
end function
```

Figure 3. Pseudocode for algorithmic insertion of an example observation

Note. Adapted from "Using ethereum smart contracts to store and share COVID-19 patient data," by Batchu S, Patel K, Henry OS, Mohamed A, Agarwal AA, Hundal H, et al. Cureus. 2022;14:e21378 (https://www.cureus.com/articles/72568-using-ethereum-smart-contracts-to-store-and-share-covid-19-patient-data#!/). CC BY.

In comparison to strings, which are dynamically sized types, storage of strings as bytes32 types—fixed-size byte arrays which consume less gas and are optimized for EVM is more economical. However, to save the IPFS hash in its alphanumeric Base58 encoding, 46 bytes are needed. Even though the hash is just 34 bytes when represented in hexadecimal form, it still exceeds the required 32 bytes. However, as previously mentioned, the IPFS identifier is also a multihash. The first two bytes can be removed because they specify the hash algorithm and length, which are universally utilized by default for the relevant present IPFS processes. This enables the storage of the hexadecimal IPFS hash as a bytes32 type (Figure 2). Data is accessible from the contract through the JavaScript console or other scripts. This allows the concatenation of the two bytes and encoding back to Base58 straightforward.

Bytes32 variables were used to hold the disease, disease grade if applicable, patient identifier, Karnofsky score, and identifier hash. Another type in Solidity is addresses, which are used to store user or contract addresses on the Ethereum blockchain.

Query

In addition to wildcards, the query tool allows you to search using up to three fields (primary diagnosis, patient ID, and Gertzbein grade). The function initially verifies which fields were queried. The corresponding values are then retrieved from the counter ID mapping using those fields as keys. All counter IDs are

retrieved if wildcards are used. The minimal length counter ID array is looped through for every record to see if the counter ID matches the counter ID in the outer loop. The struct value from the database mapping is then retrieved using the matching ID (Figure 4).

Algorithm 2 Pseudocode for querying observation

```
function QUERYOBSERVATION( STRING diagnosis, STRING patientID, STRING GertzbeinGrade)
    initialization;
    counterIDList = [ ]
    results = [ ]
    diagnoses = [ ]
    patientIDs = [ ]
    GertzbeinGrades = [ ]
    if database empty then
        return [ ]
    else
        length ← number of fields queried (0, 1, 2, or 3)
        counterIDList.push(IDs matching query)
        if length == 0 then
            counterIdList.push(all counterIDs in database)
        else
            diagnoses.push(diagnosisMapping[diagnosis])
            patientIDs.push(patientIDMapping[patientID])
            GertzbeinGrades.push(GertzbeinGradeMapping[GertzbeinGrade])
            counterIdList.push(intersection of counterIDs in diagnoses,
            patientIDs, and GertzbeinGrades)
        end if
        for i ≤ length(uniqueObservations in storage) do
            for j ≤ length(counterIDList) do
                if counterIdList[j] is ID of ith uniqueObservation then
                    results.push(database[j])
                end if
                j++
            end for
            i++
            return results
        end for
    end if
end function
```

Figure 4. Pseudocode for algorithmic querying of an example observation

Note. Adapted from "Using ethereum smart contracts to store and share COVID-19 patient data," by Batchu S, Patel K, Henry OS, Mohamed A, Agarwal AA, Hundal H, et al. Cureus. 2022;14:e21378 (https://www.cureus.com/articles/72568-using-ethereum-smart-contracts-to-store-and-share-covid-19-patient-data#!/). CC BY.

Results

With the help of JavaScript console external scripts, the smart contract was tested. Manually checking 50 random queries against the fields utilized in the queries yielded accurate results in every instance. Time and memory were also evaluated for the insertion and querying of observations in order to further analyze contract performance (Figure 5). Only 990 Megabyte (MB) of memory per insertion was required to insert 350 patient records, while retrieval required 1,026 MB. It was discovered that it took 276 milliseconds to insert 50 records and 713 milliseconds to insert 350 records. Inserting 50 records required 934 MB of memory per insertion with patient data and imaging, while inserting 350 records required almost the same amount of memory per insertion (Figure 5A). In a database of 350 records, the retrieval function needs about 1,026 MB to query a record with all three fields left blank, but only 970 MB to obtain the same observation from a database of 50 records (Figure 5B).

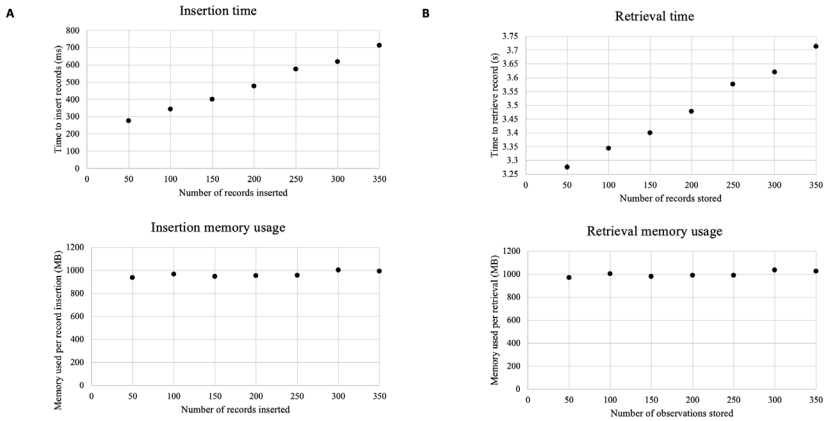

Figure 5. Smart contract performance, tested with 0, 50, 100, 150, 200, 250, 300, and 350-record inputs. (A) Insertion time and memory usage, (B) retrieval time and memory usage

Note. Adapted from "Using ethereum smart contracts to store and share COVID-19 patient data," by Batchu S, Patel K, Henry OS, Mohamed A, Agarwal AA, Hundal H, et al. Cureus. 2022;14:e21378 (https://www.cureus.com/articles/72568-using-ethereum-smart-contracts-to-store-and-share-covid-19-patient-data#!/). CC BY.

Discussion

Secure data preservation is essential within the context of ARSN as damaged data can result in medical errors. We purposed blockchain-based solutions, such as smart contracts implemented on the Ethereum platform. Such decentralized ARSN data would avoid situations in which there is a single point of failure, and create an irreversible ledger, preventing purposeful or unintentional corruption. While previous studies have demonstrated the use of this technology in healthcare [20–22], this study exemplified a smart contract integrating ARSN data. In particular, the IPFS was employed to store imaging effectively off-chain while the Ethereum blockchain was used for contract deployment. This revealed an effective way to store ARSN data while also allowing for quick storage and retrieval of observations.

However, there are limitations to implementing smart contracts as the Ethereum project is still in its early stages. For instance, building a smart contract necessitates familiarity with the Solidity programming language, which has various quirks, including the number of local variables allowed in functions and the "gas" restriction. It also requires prior platform knowledge to deploy smart contracts and set up a private Ethereum blockchain. To reduce the level of skill needed to deploy the contracts, the procedure might be written in an external script. However, there could still be challenges when communicating with the Ethereum software, such as the web3.js JavaScript application programming interface (API). The promise of adopting smart contracts for medical ARSN still exists despite these restrictions, which highlight the need for a more reliable platform. Increased patient privacy, data security, and record immutability are made possible by the integration of medical ARSN into the Ethereum blockchain platform using smart contracts.

Abbreviations

3D: three-dimensional

ARSN: augmented reality surgical navigation

EVM: Ethereum virtual machine

ID: identity document

IPFS: interplanetary file system

ISN: Image Share Network

MB: Megabyte

PoA: proof of authority

Declarations

Author contributions

SB: Conceptualization, Methodology, Data curation, Investigation, Writing—original draft, Writing—review & editing. MJD: Investigation, Writing—original draft, Writing—review & editing. KR: Writing—original draft, Writing—review & editing. LL: Writing—original draft, Writing—review & editing. BLW: Project administration, Validation, Writing—original draft, Writing—review & editing. All authors read and approved the submitted version.

Conflicts of interest

The authors declare that they have no conflicts of interest.

Ethical approval

Not applicable.

Consent to participate

Not applicable.

Consent to publication

Not applicable.

Availability of data and materials

Code is available at https://github.com/Batchu-Sai/ARSN-Smart-Contract.

Funding

Not applicable.

References

1. Appari A, Johnson ME. Information security and privacy in healthcare: current state of research. Int J Internet Enterp Manage. 2010;6:279–314.
2. Security in clinical information systems [Internet]. [cited 2022 Sep 26]. Available from: http://www.cl.cam.ac.uk/~rja14/Papers/policy11.pdf
3. Rostad L, Edsberg O. A study of access control requirements for healthcare systems based on audit trails from access logs. In: 2006 22nd Annual Computer Security Applications Conference (ACSAC'06). IEEE; 2006. pp. 175–86.
4. Anton AI, Earp JB, Vail MW, Jain N, Gheen CM, Frink JM. HIPAA's effect on web site privacy policies. IEEE Secur Privacy. 2007;5:45–52.
5. Lovis C, Spahni S, Cassoni N, Geissbuhler A. Comprehensive management of the access to the electronic patient record: towards trans-institutional networks. Int J Med Inform. 2007;76:466–70.
6. Maglogiannis I, Zafiropoulos E. Modeling risk in distributed healthcare information systems. In: 2006 International conference of the IEEE engineering in medicine and biology society. 2006 Aug 30–Sep 3; NY, USA. IEEE; 2006. pp. 5447–50.
7. Bell EA, Ohno-Machado L, Grando MA. Sharing my health data: a survey of data sharing preferences of healthy individuals. AMIA Annu Symp Proc. 2014;2014:1699–708.
8. Bitcoin: A peer-to-peer electronic cash system [Internet]. Bitcoin Project [cited 2022 Sep 26]. Available from: https://bitcoin.org/bitcoin.pdf
9. Ethereum whitepaper [Internet]. [cited 2022 Sep 26]. A next-generation smart contract and decentralized application platform; [about 1 screen]. Available from: https://ethereum.org/en/whitepaper/
10. Gürsoy G, Brannon CM, Gerstein M. Using Ethereum blockchain to store and query pharmacogenomics data via smart contracts. BMC Med Genomics. 2020;13:74.
11. Langer SG, Tellis W, Carr C, Daly M, Erickson BJ, Mendelson D, et al. The RSNA image sharing network. J Digit Imaging. 2015;28:53–61.
12. Shini SG, Thomas T, Chithraranjan K. Cloud based medical image exchange-security challenges. Procedia Eng. 2012;38:3454–61.
13. Wang C, Chen L, Chou W, Wang K. Implementation of a medical image file accessing system on cloud computing. In: 2010 13th IEEE international conference on computational science and engineering. 2010 Dec 11–13; Hong Kong, China. IEEE; 2010. pp. 321–6.
14. Hani AFM, Paputungan IV, Hassan MF, Asirvadam VS, Daharus M. Development of private cloud storage for medical image research data. In: 2014 International Conference on Computer and Information Sciences (ICCOINS). 2014 Jun 3–5; Kuala Lumpur, Malaysia. IEEE; 2014. pp. 1–6.
15. Ethereum: a secure decentralised generalised transaction ledger Berlin version [Internet]. [cited 2022 Sep 26]. Available from: https://ethereum.github.io/yellowpaper/paper.pdf

16. Benet J. IPFS - Content addressed, versioned, P2P file system. arXiv:1407.3561 [Preprint]. 2014 [cited 2022 Sep 26]. Available from: https://arxiv.org/abs/1407.3561
17. Solidity [Internet]. Ethereum Revision 7709ece9; c2016–2019 [cited 2022 Sep 26]. Available from: https://solidity.readthedocs.io/en/v0.5.12/
18. Ekparinya P, Gramoli V, Jourjon G. The attack of the clones against proof-of-authority. arXiv: 1902.10244 [Preprint]. 2019 [cited 2022 Sep 26]. Available from: https://arxiv.org/abs/1902.10244
19. Schäffer M, di Angelo M, Salzer G. Performance and scalability of private Ethereum blockchains. In: Di Ciccio C, Gabryelczyk R, García-Bañuelos L, Hernaus T, Hull R, Indihar Štemberger M, et al., editors. Business Process Management: Blockchain and Central and Eastern Europe Forum. BPM2019 Blockchain and CEE Forum. Springer; 2019. pp. 103–18.
20. Batchu S, Henry OS, Hakim AA. A novel decentralized model for storing and sharing neuroimaging data using ethereum blockchain and the interplanetary file system. Int j inf tecnol. 2021;13:2145–51.
21. Batchu S, Henry OS, Patel K, Hakim A, Atabek U, Spitz FR, et al. Blockchain and non-fungible tokens (NFTs) in surgery: hype or hope? Surg Pract Sci. 2022;9:100065.
22. Batchu S, Patel K, Henry OS, Mohamed A, Agarwal AA, Hundal H, et al. Using Ethereum smart contracts to store and share COVID-19 patient data. Cureus. 2022;14:e21378.