Blockchain: Research and Applications 3 (2022) 100066

Contents lists available at ScienceDirect

Blockchain: Research and Applications

journal homepage: www.journals.elsevier.com/blockchain-research-and-applications

# DHT- and blockchain-based smart identification for video conferencing

Morteza Alizadeh *, Karl Andersson, Olov Schelén

*Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, SE-93187, Skellefteå, Sweden*

ARTICLE INFO

*Keywords:*
Video conferencing
Blockchain
IPFS
Decentralized web hosting
Machine learning

ABSTRACT

Video conferencing applications help people communicate via the Internet and provide a significant and consistent basis for virtual meetings. However, integrity, security, identification, and authentication problems are still universal. Current video conference technologies typically rely on cloud systems to provide a stable and secure basis for executing tasks and processes. At the same time, video conferencing applications are being migrated from centralized to decentralized solutions for better performance without the need for third-party interactions. This article demonstrates a decentralized smart identification scheme for video conferencing applications based on biometric technology, machine learning, and a decentralized hash table combined with blockchain technology. We store users' information on a distributed hash table and transactional events on the distributed ledger after identifying users by implementing machine learning functions. Furthermore, we leverage distributed ledger technology's immutability and traceability properties and distributed hash table unlimited storage feature to improve the system's storage capacity and immutability by evaluating three possible architectures. The experimental results show that an architecture based on blockchain and distributed hash table has better efficiency but needs a longer time to execute than the two other architectures using a centralized database.

## 1. Introduction

Today, video conferencing and similar applications have changed a lot compared to the past. Using cloud services for storage, management, and identification solved some challenges, such as security and storage limitations on the local devices. Moreover, there is a comeback of decentralization in video conferencing as is the case with WebRTC. While decentralized video applications provide scalability in execution and storage capacity, there is a challenge to provide trustworthy decentralized identification of participants.

Machine Learning (ML) is currently used as technology in many mobile devices, such as for unlocking and identification. Face recognition has received much attention in recent years and has become the most useful and relied-on solution when combined with other Artificial Intelligence (AI) methods like voice recognition. Face recognition has become the most universally accepted authentication solution when different participants in a session do not know or recognize each other.

Among current challenges, video conferencing applications have increased access concerns by reflecting participants' information and sharing the experience with others. Therefore, identification of users and controlling the access right to data are urgent issues [1]. Additionally, applications need to be decentralized to ensure that the video conference remains a public resource that is healthy and available to all participants and that governments and third parties do not control it. Therefore, keeping the video conferencing application independent and designing it decentralized are the main objectives in this article.

Identification as one of the main challenges is discussed in this article. Finding illegal ways to join meetings, like searching and guessing available meeting links on social media websites, are significant strategies employed by uninvited attendees. All users need to prove their access rights and identify themselves as participants, which may occur during the video conference session [2]. Video conferencing application data can also be used as legal evidence for personal disputes. Therefore, how to efficiently leverage these types of data is worth studying. Further, distributed applications outsource identification to third parties (e.g., Facebook and Google) for management, which helps reduce the computational cost, saves local storage space, and provides efficient data retrieval.

Combining blockchain [3] and third parties is an approach that deals with retrieving data while providing a safeguard for data security. The blockchain system is a technology that can store event records immutably. Blockchain is a decentralized system that offers an environment for distributed applications to process and store information in an unchangeable fashion. Public blockchains have improved secure transactions or

* Corresponding author.
*E-mail addresses:* morteza.alizadeh@ltu.se (M. Alizadeh), karl.andersson@ltu.se (K. Andersson), olov.schelen@ltu.se (O. Schelén).

https://doi.org/10.1016/j.bcra.2022.100066
Received 5 July 2021; Received in revised form 17 January 2022; Accepted 12 February 2022

events in decentralized system applications due to their immutability feature [4]. Enabling a tamper-proof log of events and the distribution of data storage are typical blockchain usages [5]. However, third-party applications could imply a single point of failure in decentralized systems. This means that the system mentioned above with the third-party solution is distributed in a centralized manner because it is managed under the policy of the main party or other parties [4]. Once the third party tries to increase the subscription fee, all users must either accept or leave. Overall, applying blockchain technology to applications has become a new trend.

This article's main contribution is to present a decentralized video conferencing application that handles the identification of participants, solves the performance issue without storage limitations, and stores event logs in an immutable transactional environment. These features are chosen to obtain better performance in a decentralized manner without any third-party component. Additionally, recognizing users with the help of machine learning algorithms is a good approach for identification. In this article, the proposed solution is designed based on face recognition combined with the interplanetary file system (IPFS) [6] as one implementation of the distributed hash table (DHT) and blockchain to demonstrate the feasibility of the proposed architecture. This scheme leverages IPFS features like storage, web hosting platforms, and blockchain specialties like storing immutable event histories. Attendees can verify the data's originality and track the data source records via this system. Furthermore, blockchain addresses the effective traceability and access control of data. This combination helps ensure high accessibility, data security, scalability, trust, and flexibility.

The rest of this article is organized as follows. In Section 2, we introduce our scheme's related work. In Section 3, we present the background of our proposal. Section 4 describes video conferencing components. Section 5 provides the specific details and solution to our scheme. Section 6 evaluates the proposed method. Finally, we conclude the article in Section 7.

## 2. Related work

Video conferencing systems have changed impressively in recent years since the old systems have had several problems. Participants typically had to download an application before joining, firewalls prevented guests from joining the meeting, complexity was high when users connected through phone calls, and the meeting host sometimes needed to install software to run the centralized host meeting node [7].

Many studies have been performed on video conferencing applications in various fields, including universities, health care, and elderly care [8]. Traditional video conferencing applications face several of the above-mentioned problems. Agrawal [9] showed that the cloud can increase performance and foster a new age in this area. MS Teams and Zoom are two examples of these efforts. Okereafor and Manny [10] explained how to deal with security issues in serverless video conferencing applications. Additionally, identified participants could join video conferences, as such an application should be open for all to join. Mohanty and Yaqub [11] have focused on participant identification and authentication. Dargan and Kumar [12] and Elgharib et al. [13] found that there is a need for better applications and that new applications have acceptable features using Machine Learning algorithms for identification and authentication, like face recognition, fingerprinting, and IRIS. Additionally, it is obvious in the security area that using blockchain and other decentralized systems like IPFS has increased efficiency [14]. The combination of video conferencing, IPFS, blockchain, and ML upgrades older systems so that they can achieve better performance. It also improves security and solves storage and user identification issues without installing several different components. Recently, several similar works used blockchain and IPFS in their studies with different aspects concerning image copyright, preserving privacy, and agricultural smart contracts [15,16].

Mubashar et al. [17] provided data availability by considering how data are stored in IPFS, how IPFS deals with data permanence, and the advantages and disadvantages of IPFS's storage system. Also, they mentioned that they improved the entire system's computational complexity and processing power by using quantum computing and quantum mechanics. Furthermore, they mentioned IPFS as a solution for storing data in a decentralized fashion, which is one of the same issues this article is trying to improve. Moreover, this article tries to improve access to the application with the help of ML and image processing combined with a trackable public blockchain solution for storing the access history as an event in a permissionless public blockchain environment.

Current related works and their main methods are summarized in Table 1.

## 3. Background

Our proposed video conferencing application has five main technologies and six properties that can be mapped into different areas. In the following, these areas will be explained briefly.

### 3.1. Technologies

Technologies in computer science are changing from concentrated to distributed modes of production and applications. Technology includes applications, skills, techniques, and processes that accomplish goals in

**Table 1**
Recent researches.

| Related Works | Year | Title | Overview |
|---|---|---|---|
| Agrawal [9] | 2021 | A survey on recent applications of cloud computing in education: COVID-19 perspective | Cloud can increase performance |
| Okereafor and Manny [10] | 2020 | Understanding cybersecurity challenges of telecommuting and video conferencing applications in the COVID-19 pandemic | Security issues in serverless video conferencing applications |
| Mohanty and Yaqub [11] | 2020 | Seamless authentication for online teaching and meeting | Participant identification and authentication |
| Dargan and Kumar [12] | 2020 | A comprehensive survey on the biometric recognition systems based on physiological and behavioral modalities | Machine learning algorithms and 3D biometric algorithms |
| Elgharib et al. [13] | 2020 | Egocentric videoconferencing | Enables handsfree video calls for video conferencing application |
| Alizadeh et al. [14] | 2020 | Efficient decentralized data storage based on public blockchain and IPFS | Decentralized solutions for better efficiency |
| Kumar et al. [15] | 2021 | SP2F: a secured privacy-preserving framework for smart agricultural unmanned aerial vehicles | Blockchain, and smart contract-based enhanced Proof of Work |
| Kumar et al. [16] | 2021 | A secured distributed detection system based on IPFS and blockchain for industrialimage and video data security | Detecting copyright in images |
| Mubashar et al. [17] | 2021 | Storage and proximity management for centralized personal health records using an IPFS-based optimization algorithm | Providing data availability by using blockchain and IPFS |

both public and private eras. Concepts of decentralized technology are used throughout all technology types, including video conferencing, machine learning and face recognition, identification, decentralized platforms, IPFS, and blockchain. In the context of computing and information technology, a decentralized system contains a group of electronic devices that are connected. Usually, they take the form of networked computers like peer-to-peer (P2P) networks. These systems are connected, and no single entity is the sole authority. Decentralized platforms attempt to keep the data out of companies' servers and seldom manage data under the user's control. IPFS and blockchain are two popular decentralized technologies.

#### 3.1.1. Video conferencing technology

Video conferencing involves a technology to communicate via electronic devices and allows users to continue face-to-face meetings in different locations without gathering in a single location. This technology is especially appropriate for academic and business users in different places to save time, expenses, and hassles associated with travel.

#### 3.1.2. Machine learning and face recognition

AI is intelligence demonstrated by machines. It is a simulation of human intelligence in machines that makes machines learn from experience, adjust to new inputs, think like humans, and mimic their actions [18]. Machine learning is a sub-category of artificial intelligence. It is an area of computer science in which a computational method is trained to act automatically and make decisions by its memorized knowledge. It can learn from data and identify patterns automatically, such as supervised and unsupervised techniques without direct human intervention. It is designed based on the idea that systems can learn from their experiences or history. Machine learning has given us self-driving cars, face and voice recognition, and intelligent web search platforms like MSN, Bing, and Google. Face recognition is one application of machine learning technology. It works based on matching the data on a human face from a taken image or video from a camera against those of several stored faces. It can be used as an authentication technology and works by extracting and measuring face features from a given image.

#### 3.1.3. Identification

Identification in computer science is a process of naming entities. Identities help systems verify that a person is who he or she says. National identity numbers are examples of identification inside a border.

#### 3.1.4. Distributed hash table

IPFS is a decentralized system that is designed based on distributed hash table technology [19], with no single point of failure. IPFS is a decentralized storage protocol designed to address excessive file redundancy and allocates a unique hash for each stored file. The user can retrieve the corresponding file according to the unique hash address. IPFS helps store decentralized data without memory capacity limitations [14].

Furthermore, IPFS is a decentralized platform for sharing files. Additionally, it covers a DHT to track who holds what data [20]. IPFS has command line and graphical user interfaces that make it uncomplicated for users to operate. It stores data packages by providing blocks of data. However, the distributed part of DHT means that the entire table is spread across different locations [6]. IPFS uses Kademlia technology to recognize which nodes have which data. Kademlia is defined based on DHT for decentralized peer-to-peer computer networks and was designed by Maymounkov and Mazieres in 2002 [20]. IPFS can store data without considering the size of the data. Users can then use the hash address of the recorded data to retrieve the data. The stored data will be divided into many small chunks, where each chunk is identified with its hash address. These chunks are distributed among neighboring nodes that have their hash closest to the node. Finally, all the chunks can be combined to recreate the main object after visiting all the small chunks when a query request is received by the system.

#### 3.1.5. Blockchain

Blockchain is a decentralized system. It presents a context for distributed applications to process in an immutable way [4]. Storing a cryptographic signature of recorded data and events is one of the main features of blockchain. Furthermore, it addresses how to keep transactional data unchangeable. Permissioned and permissionless blockchains are two types of blockchain technology in decentralized systems [21]. Permissionless blockchain means that it is an open system for all users to join or delete their accounts. The cryptocurrency concept (like Bitcoin) is a famous example of a permissionless blockchain. It operates the decentralized digital currency while removing central management. The blockchain contains sets of data packages called blocks that represent the transactions' historical data. Timestamps and hash values are the main block features that are unique. Each block is chained to the previous block, which is known as a parent block. It is possible to return to the initial block or the genesis block by tracking the parents. A new block is added when most network parties accept it by their specific consensus mechanism [21]. The data will be published across several networked parties. All nodes that received the data will replicate and preserve an exact copy of the transaction information after the consensus operation is finished. This information is stored on each node separately and causes trust among them. It is essential to pay for operations and transactions with money or credits. These credits motivate parties to participate in consensus, referred to as proof of stake or proof of work [4], and earn money from the commission of executing the transaction. Furthermore, network participants are motivated to compete in a competition to earn more credits. Ethereum is one of the permissionless blockchains that works based on smart contracts, where Ether is its cryptocurrency. A smart contract is a computer program or a transaction protocol based on the Solidity programming language. It can execute automatically and perform legally relevant events and actions according to the terms of a contract or agreement [14].

### 3.2. Properties

Various properties are the outcome from a decentralized perspective. Some of the popular properties are selected to be explained in the following sections.

#### 3.2.1. Accessibility

Today, electronic devices are equipped with fast and robust hardware. Accessibility refers to the accessibility of an application to all people's electronic devices. The term accessibility is most often used in reference to specialized hardware or software or a combination of both, designed to make it easy for users. Applications should be designed for easy usage to make the use of technology less challenging for those with disabilities.

#### 3.2.2. Data security

Data security means protecting digital information from unauthorized access and malicious activity during its entire lifecycle. Ideally, information must be protected by encryption, data masking, and a reduction in sensitive file regulation. It emphasizes all aspects of information security from hardware, software, storage devices, access control, and the security of software applications.

#### 3.2.3. Scalability

Scalability is the ability of a system to manage a massive number of tasks by allocating resources like computers, networks, programs, and applications to the system. Video conferencing applications are one example that typically should support an increasing number of users and manage all users with the best performance and lowest response time.

#### 3.2.4. Trust

A trusted application is protected from modification by untrusted components with hardware. Users do not trust a system with uncertainty, and this can evoke feelings that the application cannot be relied upon or will not turn out to be good. In particular, such feelings convey being unsure about something, even if it cannot be proven. Video conferences, as an example, must be designed well so that users feel that live video communication and document sharing are safe by accepting the applicable policy.

#### 3.2.5. Flexibility

Flexibility in computer science means that applications can join and cooperate with other applications or side applications. A flexible, well-designed application can be easily used and connected to other modules. Additionally, updating these applications is not complex.

#### 3.2.6. Performance

Performance analysis is the systematic observation of the action or process of performing a task or function. The performance of applications depends on various parameters. Application performance is calculated based on user needs and application response time. High performance can be achieved when our application provides high accessibility, immutability, scalability, trust, and flexibility.

## 4. Video conferencing components

Today, video chat applications typically use IP communication. Centralized Teams, WhatsApp, and Zoom are popular applications among people. Decentralized Web Real-Time Communication (WebRTC) and Jitsi are free and open-source technologies that provide web browsers and mobile applications with real-time communication features [22]. They allow developers to build simple video conferencing programs through application programming interfaces (APIs) using HTML and JavaScript, eliminating the need to install plugins or download native apps. This communication is extremely fast with low latency, as the connection is P2P, and there is no server in the middle. Google developed this technology, and currently, almost all browsers support it [22]. Fig. 1 illustrates web conferencing using WebRTC.

### 4.1. WebRTC signalling mechanisms

WebRTC supports a variety of signalling mechanisms. Signalling is the needed data exchange to set up, change, and tear down a connection. Signalling is the center of peer detection and supports establishing communication among users by exchanging data through channels [23]. Signalling performs communication among browsers and other peers with a specific server or servers. Furthermore, signalling supports multiple protocols to combine the network addresses and port numbers in the media exchange area. Additionally, Session Traversal Utilities for NAT (STUN) and Traversal Using Relays around NAT (TURN) are two methods that WebRTC supports for Network Address Translation (NAT) traversal [23].

### 4.2. Jitsi technology

Jitsi is a multiplatform, free, open-source Voice over Internet Protocol (VoIP), video conferencing, and instant messaging application framework. Jitsi allows developers to build and deploy video conferencing applications quickly. Jitsi Video Bridge and Jitsi Meet are two significant components that help developers implement conferences on the Internet. In addition, the Jitsi team developed an ad hoc traffic generator dedicated to testing the performance of their open-source video bridge modules [24]. Failure to transmit videos is one of Jitsi's internal problems that has occurred when there are many peers in the tests [24].

### 4.3. WebRTC and Jitsi advantages

All major browsers support WebRTC and Jitsi technology. They can also be bundled into native applications, and users do not need to install any additional software to utilize the video conferencing application. The applications can be customized and combined to include multiple tools for business needs. Some of these incorporated tools, such as whiteboards, sharing, and chat, can be used in online education and medical devices. WebRTC and Jitsi encrypt all the communication data that they send among participants. The HTTPS protocol grants security to access media devices like cameras and microphones through communication. This standard ensures that only secure connections can be set up.

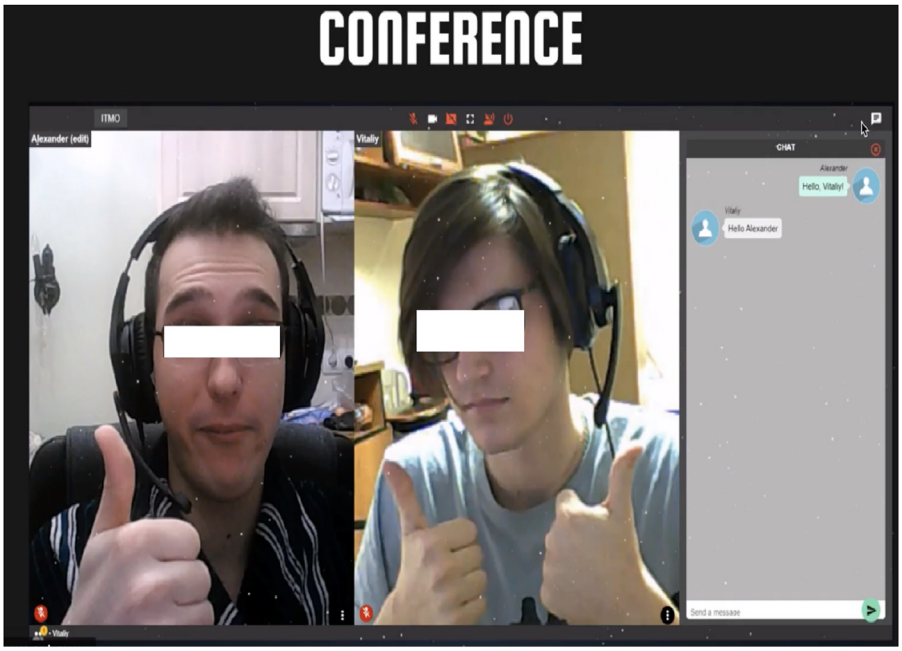


**Fig. 1.** Web real-time communication-based video conferencing.

### 4.4. Face recognition

Face recognition is a method of identifying a person by a provided image of his/her face. A face recognition system is a technology capable of matching a human face from a digital image or a video frame against a database of faces, typically used to authenticate users through ID verification services and works by pinpointing and measuring facial features from a given image. The face recognition process starts by taking an image for each person associating the file with a name or identity. Then, it compares the input image to the reference data in the database and finds the most similar reference image. The user is recognized by the system when both images are similar enough. Face-api.js is a famous face recognition library that is a JavaScript module built on top of the TensorFlow.js core. Face-api.js implements a simple Convolutional Neural Network (CNN), which returns the landmarks of 68 points of the user's token face image. This method compares taken pictures with face descriptors of the reference data. Then, differences between two face descriptors will be determined based on a threshold value [25,26].

It builds up multiple convolutional neural networks to match images and perform face recognition automatically. Several models are available with face-api.js, including face detection, facial landmark detection, face recognition, facial expression recognition, age estimation, and gender recognition [27].

### 4.5. Problems

Current video conferencing applications have some common problems. The problems are divided into three main categories: the first category concerns the identification problem, the second category concerns the immutability of transactions, and the last category concerns the risk of failure in centralized systems. Identification and immutable transaction issues are considered major problems of decentralized applications. Therefore, video conferencing APIs should be designed well to guarantee these aspects for their users. Although machine learning is a smart technology that can help solve the identification issue, the transactional aspect still remains to be addressed. The next section illustrates possible answers to the three abovementioned issues in terms of video conferencing, which is the main focus of this article.

## 5. Proposed solution and architecture

We propose a new identification system for video conferencing applications. We evaluate the proposed solution in three scenarios with different aspects to explain more about the different features of our proposal.

### 5.1. Scenarios

The first scenario is face recognition for identification. Smart face recognition is known as the best solution because it uses media devices for video conferences. It is also possible to ask the participants to show their faces when logging in to the system. The best solution for identifying a user is to use machine learning methods to extract the face and recognize who is running that device to join the video conferencing application. Neural networks technology helps compute a face descriptor (a feature vector with 128 values) from any given face image. Additionally, face-api.js uses multi-task cascaded convolutional neural networks (MTCNNs) as an algorithm that considers five-point facial landmarks to detect faces in an image [28]. This method shows a good result, but since malicious users or attackers can steal facial descriptions, it remains risky.

The second scenario is using IPFS as the host. IPFS helps run serverless video conferencing, and there is no need for a centralized server to execute it, where storing bulky data in a decentralized manner is its main feature. OrbitDB is a serverless, distributed, peer-to-peer database [29] that uses IPFS as its data storage and IPFS's Pubsub mechanism to automatically sync databases with peers. It is ultimately a consistent database that uses CRDTs, where a CRDT is a data structure that can be replicated across multiple computers in a network. The conflict-free database merging feature makes OrbitDB an excellent choice for decentralized apps (DApps) and blockchain applications. It has a better result when comparing it with the first scenario. It is too difficult to guess the permission key to access the database and facial description file, but the risk of updating the database by attackers remains. The third scenario involves combining blockchain and DHT technology. IPFS stores files in a decentralized fashion. OrbitDB is not immutable if someone steals a permission key to access the database. Blockchain can be useful here. It stores the access and address of people who can edit or remove databases on the blockchain. Additionally, it stores events and logs with timestamps. Also, all network members know who accessed the database on different occasions. This scenario exhibits better performance than the previous scenarios. However, the complexity of this model is greater than that of the previous scenarios. It is difficult to guess the permission key, generate fake facial descriptions, and guess blockchain wallet addresses, but the risk of updating the database by attackers remains. Table 2 shows a performance comparison among these scenarios.

### 5.2. Parameters and comparison

Data immutability is one of the main issues. Keeping data in computers, electronic systems, and networks unchangeable is the meaning of immutable data. In the case of video conferencing, immutability means keeping data like attendees' login information, ID, address, and timestamp immutable. Typically, blockchain is used as a solution for keeping an immutable global event or transaction ordering. It protects storage resources from accidental or intentional destruction and unauthorized or malicious user access to open and public systems. It is not possible for any party to change this information because everyone stores a copy in their memory. Blockchain presents trust and immutability without central authorities' help.

A smart contract is stored on a blockchain and is automatically performed when any user in the network wants to execute it. In this article, we use smart contracts to log transactions in a serialized and immutable way. We also control and manage the relevant events.

Decentralization is the next issue, as it means that a system is not owned, controlled, and managed by a single person or authority. In this article, decentralized means that the system should be designed so as not to have a central unit or any third party to manage or control users' activities with the lowest amount of failure risk.

Table 3 shows a comparison between centralized and decentralized video conferencing applications. Users expect the video conferencing application to be easy to install, with no third-party interference, no communication failure, and no potential for security problems.

For example, some applications like Zoom can be installed or have a web application that users can use. A decentralized system means that if some failure occurs on the application server or servers, then users can continue communication through another server. We need an application running as a decentralized application on top of the decentralized network. Additionally, video conferencing APIs need to be decentralized in the data storage system. Recording video, sharing documents and files, and editing presentation slides are the main requirements of a video

**Table 2**
Performance comparison.

| | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| Accessibility | Low | High | High |
| Data Security | Low | Medium | High |
| Scalability | Low | High | High |
| Trust | Low | Medium | High |
| Flexibility | Low | High | High |
| Login Process Time | Low | High | High |

**Table 3**
Comparison among systems.

| Item | Typical Video Conference | Our Proposal |
|---|---|---|
| Need to connect to database/use cloud for storage | Cloud-based database | Distributed database |
| Applications being centralized/ decentralized | Distributed centralized | Decentralized |
| Users must trust 3rd-party components | Yes | No |
| Identifying users by 3rd-party/consensus | 3rd party | Consensus |
| Some failures in servers cause the halting of the whole system | Partial failure | No |
| Peer-to-peer communication | Yes | Yes |
| Peer-to-peer file sharing | No | Yes |

conferencing application. Recently, IPFS has started to meet these requirements. Developers can upload and compile their code on IPFS as a decentralized web hosting node.

Furthermore, Ethereum, as a popular blockchain service, has started to sell domains for web APIs known as the Ethereum Name Service (ENS). It functions analogous to a domain name service, and it supports IPFS links to publish it in its domains. This process helps shorten web domain names compared to the IPFS hash address.

### 5.3. Architectures

Fig. 2 shows an overview of how users can be verified by our proposed methods and illustrates the overall view of the proposed method.

In this view, users interact with the system through four main modules. Face recognition is the first module that includes multiple libraries. It helps the application identify users. This module's extracted faces should match the stored faces from the database for the validation process. IPFS is the next module that has the storage role and can also support multiple roles, like being a web hosting node, and by compiling JavaScript code. Identified users must be eligible to communicate with the video conferencing application, as in the other modules. The blockchain module stores information in its blocks, like the addresses of logged-in users, events, and timestamps, as a transaction ordering process after the face recognition module has validated them. A public blockchain is regarded as a solution to keep information immutable in a decentralized system. Although a lack of memory for storing data decentralized with different sizes concerns the blockchain issue, IPFS can overcome these problems.

Fig. 3 illustrates the architecture layers. In the first layer, users communicate with the face detection API in the second layer. Face detection is also executed by an IPFS web hosting node that communicates with a decentralized file system layer as a third layer. Finally, the sub-modules should interact with the blockchain as the last layer for the transaction processing and storing of events. This architecture can protect the whole system against malicious users. The following section explains three possible architectures, which are evaluated in the following section. The first architecture implements simple face recognition functionality with a centralized server. The second architecture shows a system without IPFS, while the third architecture is an improvement of the second architecture in terms of handling and managing decentralization.

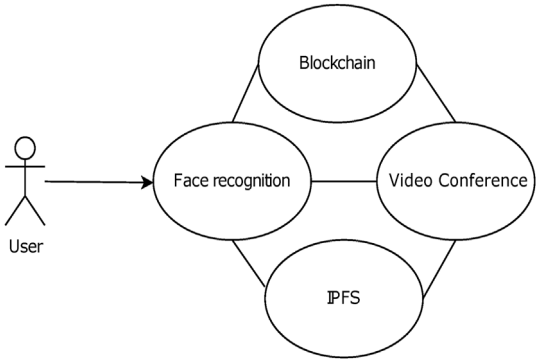


**Fig. 2.** Overall view of the architecture.

#### 5.3.1. First architecture (1st Arch)

In the first architecture (1st Arch), as the traditional style of video conferencing applications, we consider that the system has a centralized face recognition service and uses a central database. Additionally, the video conferencing program is loaded on this server. In this model, all users communicate with the central server, and users can log in to the system after the face recognition process has finished.

#### 5.3.2. Second architecture (2nd Arch)

In the second architecture (2nd Arch), as shown in Fig. 4, users want to communicate through the network. The data object looks similar to a request holding all the information needed for face recognition. Moreover, web hosting technology helps the system store records of data in its database. It sends accepted user requests to the blockchain to store event logs after the recognition module has validated the user. After the announcement and publishing of the information as a block, other network members can see the information. The transaction is a process that shows all the abovementioned steps by storing the information on the blockchain. This transaction stores all the information, which includes user signatures, addresses, timestamps, and smart contract terms. Fig. 5 demonstrates a smart contract for storing data. It includes a class that has a data structure and functions to record data on a smart ledger. This information will allow the network to leverage it to prevent some protection naturally against security threats. This architecture cannot help improve the centralized storage problem because blockchain has limitations in storing bulky data.

#### 5.3.3. Third architecture (3rd Arch)

In the third architecture (3rd Arch), as shown in Fig. 6, users want to communicate through the network. The rest of the process is similar to the 2nd architecture. The main differences are that there is an IPFS web hosting service, and further, it covers storage issues without any central unit or third-party interaction. The system can also record bulky data in a decentralized fashion by following this idea.

### 5.4. Summary of architectures

Several significant parameters are essential in the system: 1) time, 2) immutability, 3) risk of failure, and 4) performance. In this article, performance is defined based on immutability and risk. A system delivers high performance when executed in a decentralized environment with the lowest probability of failure. Fig. 7 shows an overview of the three architecture performance analyses based on the three mentioned parameters. The 1st architecture is a traditional centralized system, while the 2nd and 3rd architectures are the second and third mentioned architectures, respectively.

The 3rd architecture performance is the best compared to the others because it is immutable and IPFS helps this system reduce the risk of failure. Both the 2nd and 3rd architectures contain many functions that validate and distribute the ledgers in the blockchain. Thus, these systems are slow. The 1st architecture is the fastest, but the risk of failure is higher than that of our decentralized systems.

## 6. Evaluation

In this section, the 3rd architecture is discussed as the main approach in four steps: 1) face recognition, 2) IPFS, 3) Jitsi, and 4) storing data in the blockchain.

OrbitDB version 0.24.2 is used as a database. JavaScript (npm 11.7.2 and node.js 14.16.1) is selected as the programming language. Ethereum

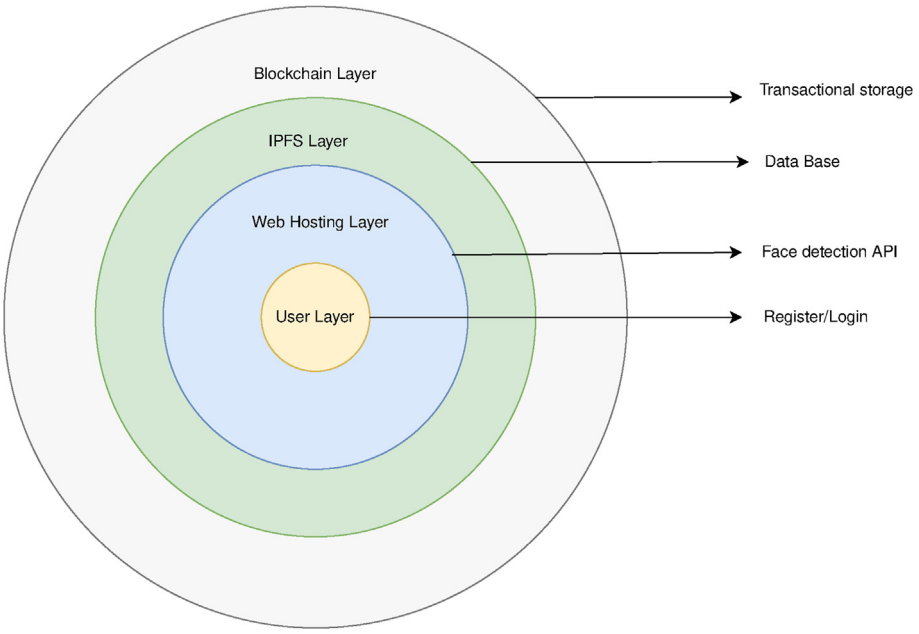


**Fig. 3.** Architecture layers.

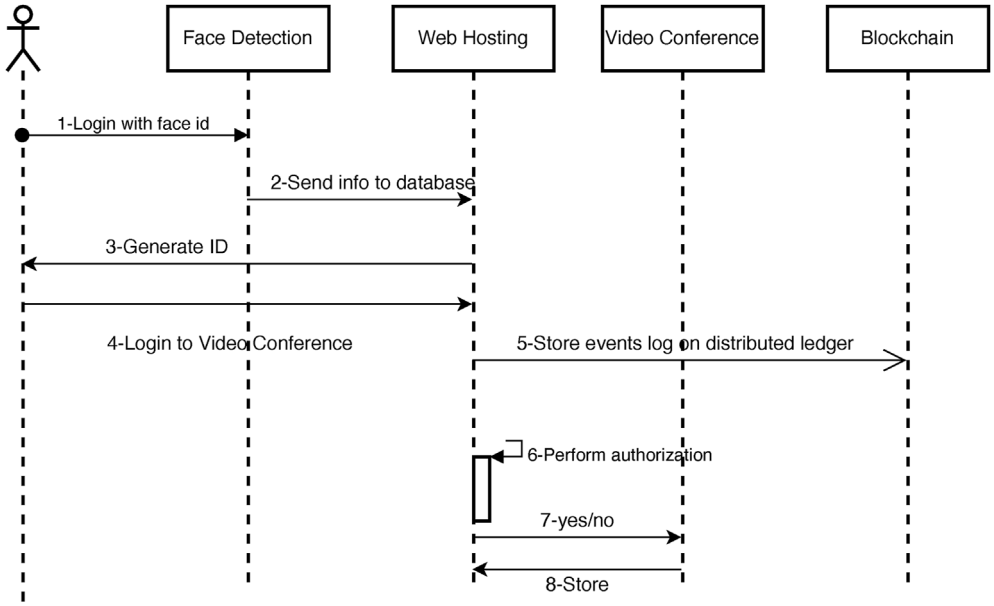


**Fig. 4.** 2nd architecture.

is a popular public blockchain, and Remix IDE is a famous editor for compiling and running the smart contracts written in the Solidity language in Ethereum. Metamask is an Ethereum wallet that includes accounts, addresses, and credit (ETH). Credit is necessary to sign up and then connect it to the blockchain. We used the Ethereum Kovan test network for testing smart contracts in a public blockchain.

We repeated the experiment 10 times. The results are shown in Figs. 8–10 and Table 4, where the average for the abovementioned functions is indicated in each caption. We used a MacBook Pro (15-inch, 2018) laptop with the following configuration: macOS Big Sur Version 11.0.1, 2.2 GHz 6-Core Intel Core i7 processor, 16 GB 2400 MHz DDR4 memory, 95/91 Mbps average download/upload data rates, Google Chrome Version 90.0.4430.212, IPFS Version 0.3.11, face-api.js Version 0.17.0, and react-jitsi Version 1.0.4.

Fig. 8 gives the average time for face recognition components and computing times for seven different procedures. The Get, Query, and Upload times are displayed in relation to fetching data from local storage, whereas information about the database is illustrated in Fig. 10. From the information in Fig. 8, we can see that the face recognition function duration is the highest at approximately 6.75 ms and the lowest duration in the Get function (import face elements) is 0.00012 ms. The total average time is the sum of the times needed to import an image, extract the face, and check the similarity. The average total time is 8.20 ms.

Fig. 9 shows ten repetitions of running the Create and Query functions of the blockchain module. The Query time covers fetching data from the ledger, while the Create time represents executing transaction time, which is the time for performing a new transaction to add a new record to the smart ledger after consensus by Ethereum. The Create time is greater than the Query time, which is illustrated in Fig. 9. From the information in Fig. 9, we can calculate average times. The Create function is the highest at

```
pragma solidity ^0.5.0;

contract Eventlog {
    string public name;
    uint public Counter = 0;
    mapping(uint => Record) public records;

    struct Record {
        uint id;
        string name;
        uint256 time;
        address payable owner;
        bool recorded;
    }

    event RecordCreated(
        uint id;
        string name;
        uint256 time;
        address payable owner;
        bool recorded;
    );

    constructor() public {
        name = "Our Smnart Contract";
    }

    function createRecord(string memory _name, uint256 _time) public {
        // Require a valid name
        require(bytes(_name).length > 0);
        // Require a valid timestamp
        require(_time > 0);
        _time = now;
        // Increment Records count
        Counter ++;
        // Create A Record
        records[Counter] = Record(Counter, _name, _time, msg.sender, false);
        // Trigger an event
        emit RecordCreated(Counter, _name, _time, msg.sender, false);
    }
}
```

**Fig. 5.** Smart contract for recording events.

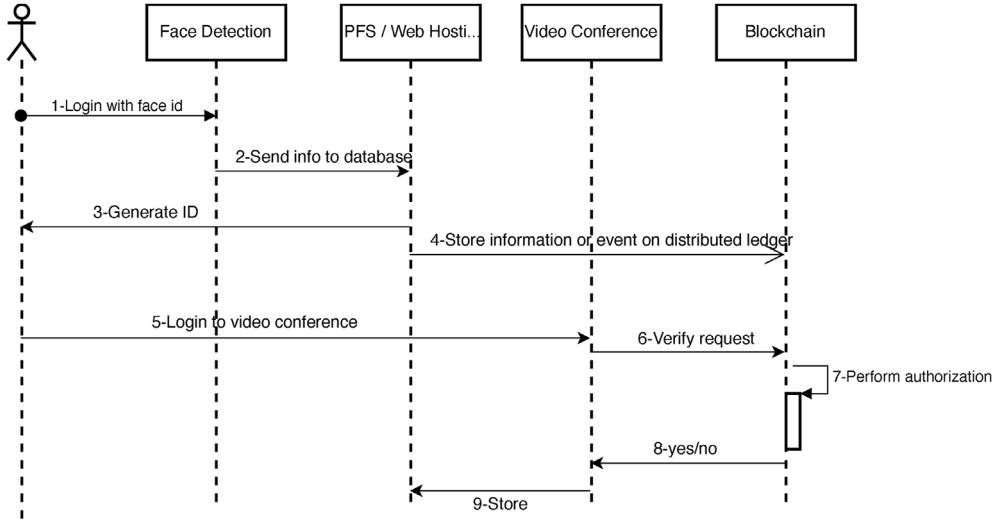


**Fig. 6.** 3rd architecture.

approximately 466.3 ms, and the lowest average in the Query function is 6.217 ms.

Fig. 10 shows ten repetitions of running the IPFS functions and the OrbitDB modules. The Insert times refer to uploading data to OrbitDB. The Get times refer to fetching data from the OrbitDB values. The Connection time is the time to connect to OrbitDB on top of the IPFS node. The Upload times are the sum of the Get and Insert times. The Query times are the sum of the Get times and Connection times. The Extraction times are the time to

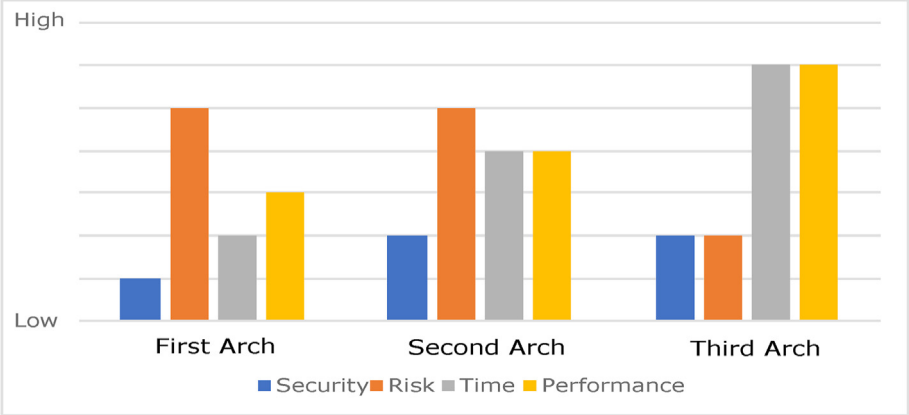


**Fig. 7.** Performance based on immutability and risk.

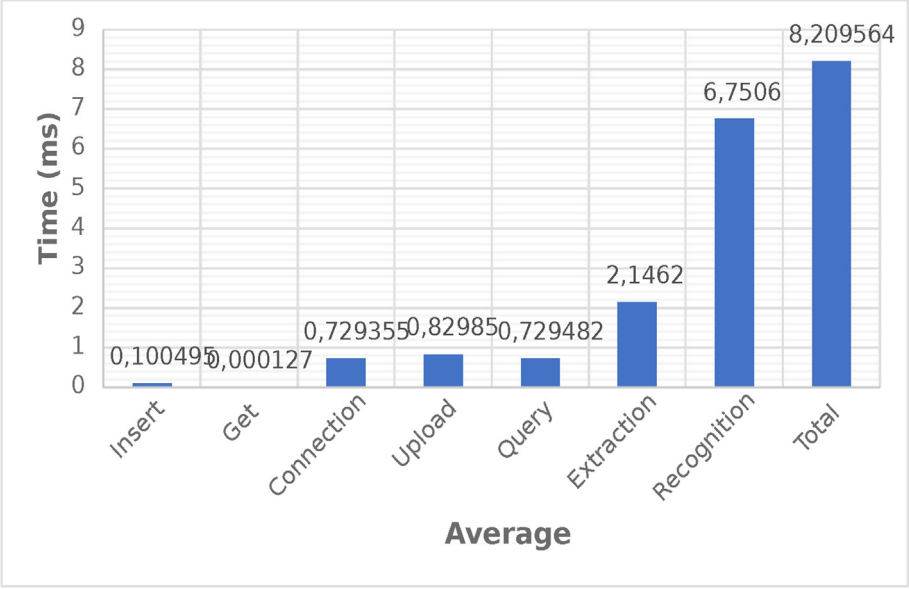


**Fig. 8.** Average times for face recognition component.

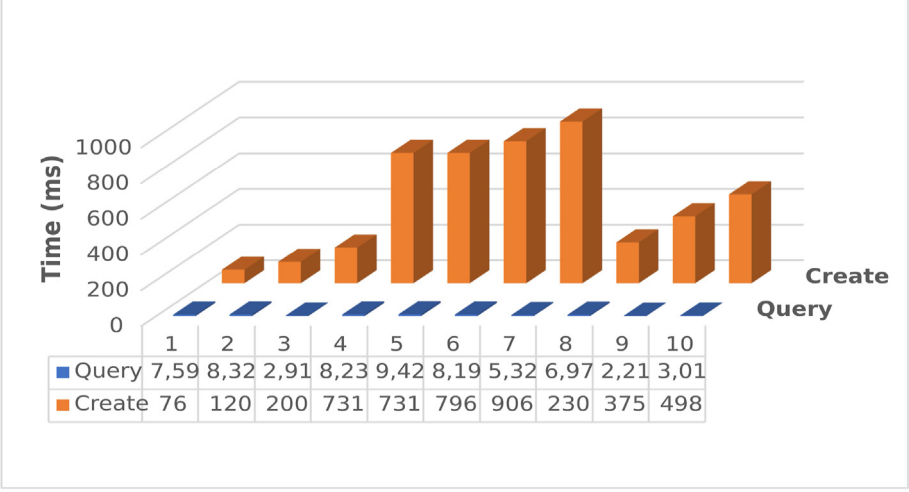


**Fig. 9.** Average blockchain process times.

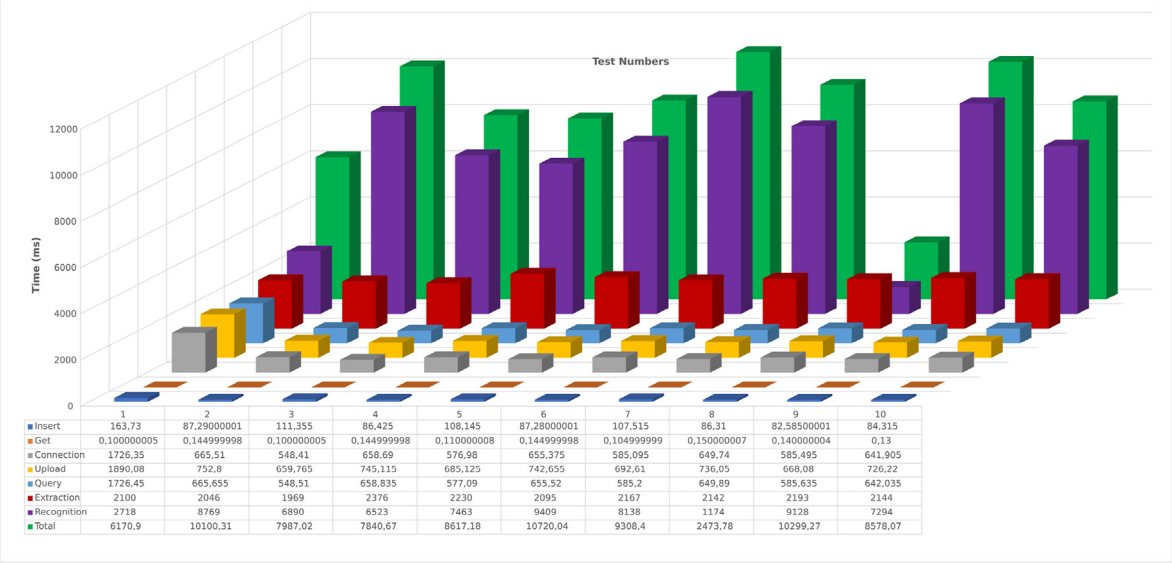


**Fig. 10.** Performance based on waiting time for login.

**Table 4**
Serverless and server-based system comparison.

| | Localhost | IPFS-P2P |
|---|---|---|
| Requests | 9 | 9 |
| Requests to Host | 7 | 7 |
| Total (ms) | 405 | 421 |
| Time to First Byte (TTFB) (ms) | 7 | 115 |
| DOM Content Loading (ms) | 194 | 254 |
| Slowest Call (ms) | 53 | 119 |
| Average Call (ms) | 28 | 24 |

extract facial features, which is the same as the extraction times in Fig. 8. The Recognition times are the most significant times to recognize faces and find matches from the database. The total average times are the sum of the collected times of the Get, Connection, Query, and Recognition functions to import an image, extract the face, find matches, and determine the similarity from the IPFS-based database. The average total time for the IPFS module is 8209 ms.

Table 4 shows a comparison between two video conferencing models running on the localhost (centralized host) and the IPFS WebHost (decentralized host). This table shows the average of ten repetitions to measure the request numbers to host and sub-domains. The total time shows the waiting time to log in to the video conferencing application. The critical parameters here are the Total time and Average Call time, which show that the localhost has better speed than the decentralized model.

### 6.1. Results and discussion

In the previous sections, the third architecture is discussed as the main approach in four steps: 1) face recognition, 2) IPFS, 3) video conferencing application, and 4) storing event log information in the blockchain.

The first step is face recognition, which is built based on Node.js. The input of this step is a captured image. A face extractor function from a face API library extracts a face from the captured image and creates a face description file. It uses a simple form of search algorithm by comparing new description files with stored information. An identified user is a user when the search function finds a similar captured face description file in the database. The second step is IPFS. It solves storage and hosting problems in a decentralized manner. IPFS can compile JavaScript code, and in this case, it is used as a host to upload the code and has no limitations for storing data. There are two modes for using IPFS: the IPFS desktop version or command line version. The desktop app has an easy-to-use interface for adding, pinning, and sharing files. The output of this step is a hash code that shows the address of the compiled decentralized application. This code is too long, and it is difficult for users to remember. ENS is a technology-based Ethereum blockchain that solves the problem by providing a decentralized name service for IPFS applications. The third step is video conferencing, which uses WebRTC and Jitsi technology for communication. Additionally, this module can be compiled by IPFS. Moreover, the WebRTC and Jitsi video conferencing applications have a module for text chatting and screen sharing. The last step is blockchain. Ethereum is a technology that helps store data immutably. Fig. 5 illustrates a smart contract to store event information like name, address, and time in Remix IDE. All published smart contracts have a specific address, and it is enough to connect a project to this smart contract by knowing the smart contract's address. After the validation process has been completed, the blockchain can add new transaction data to its immutable historical records.

The total time to execute the whole application is 8638.2 ms, which is the sum of the face recognition process time (8.2 ms) plus the time to communicate with OrbitDB (8209 ms) plus the time to log in to the video conferencing application (421 ms). Algorithm 1 illustrates the functions and script sequence to be compiled by IPFS. In the initialization part, the application needs to be uploaded to IPFS. The IPFS compiler loads the dependencies to compile the project. Then, the main process starts by taking a face photo, checking the similarity or identifying, and sending a request to the blockchain to store the event, and finally, the abovementioned process finishes by connecting the user to the video conferencing application.

The result shows the long waiting time of OrbitDB that made the decentralized model slow. In this article, performance is emphasized on decentralization and immutability, whereas the proposed architecture has a better result. If we consider the time to evaluate the performance, the centralized architecture is the best. Otherwise, the decentralized architecture is better than the centralized architecture because there is no centralization, and the risk of failure is less than for the centralized architecture. The blockchain process time is not high compared to that of IPFS. Thus, the time does not affect the sign-in time of video conferences.

Our analysis refers to our implementation as linearithmic computational time complexity, where $T_1$ is the time complexity of the face landmark detection algorithm based on the number of pixels that images have, $T_2$ is the time complexity of face recognition's functions and finding the match, and $n$ is the number of face descriptions in the database, $T_H$ is the time complexity of reading data from DHT (time to look up), $T_{up}$ is

the network connection time that is independent of algorithms that we used, and $T_{BC}$ is the blockchain transaction execution time that consists of independent parameters like Difficulty. For example, Difficulty in the Ethereum network is the difficulty of a problem that miners must solve to find a block. It means high Difficulty needs a longer time to be solved. $T_{total}$ is the sum of all previous processes. Table 5 summarizes the time complexity of all the above processes.

Blockchain modules consume many resources, so the computational and operational transaction costs are dependent on many parameters, like the number of nodes and type of consensus. Therefore, there is some research to measure the complexity of a private and consortium blockchain.

**Algorithm 1**. Process of Smart Identification for Video Conferencing Application

**Algorithm 1:** Process of Smart Identification for Video Conference Application

```
Run Dependencies
{Node.js, Orbit-db, IPFS, Blockchain};
IPFS-compile(Face Recognition(), Video Conference Application());
while user not found do
    TakePhoto();
    Accept Blockchain Transaction();
    if faceID matches with database then
        user.information = fetch info.Get(faceID);
        Start(Video Conference(user.information));
        blockchain Create(user.information) ;
    else
        return "user not found";
    end
end
```

## 7. Conclusions and future work

The security aspect is necessary to deal with applications that help satisfy the digital world's expected requirements. Identification issues and immutable data record systems should therefore be given attention. This article presented a combined machine learning face recognition solution based on a public blockchain and IPFS. We started the article by explaining video conferencing applications, blockchain, and smart contract solutions. Furthermore, we discussed how the blockchain manages transactions by its immutable historical data. Supporting decentralized web hosting and sharing data in a DHT or IPFS were discussed. This system can be beneficial for identification among decentralized systems. The IPFS decentralized web hosting solution is also considered a great solution to keep all systems decentralized without having to worry about memory limitations. Finally, this article illustrated two new architectures for video conferencing combining blockchain and IPFS. We showed that the proposed solution is feasible because of its decentralized and immutable features.

Although the blockchain is a good solution to keep transaction data immutable, the risk of new security attacks remains. For future work, we consider adding new methods for strong identification by entering all user agreements and securing the system against new security threats. We also consider extending our study to cover multi-user identification and user privacy issues.

**Table 5**
Time complexity.

| Time | Description | Complexity |
|---|---|---|
| $T_1$ | Complexity of face landmark detection algorithm (where $n$ is the number of pixel) | Linearithmic time |
| $T_2$ | Complexity of face recognition's functions and find the match (where $n$ is the number of faces in the database) | Linearithmic time |
| $T_H$ | Complexity of reading data from distributed hash table (DHT) (where $n$ is the number of nodes) | Logarithmic time |
| $T_{up}$ | Network connection time (Internet speed) | Independent constant time |
| $T_{BC}$ | Blockchain transaction execution time | Independent constant time (ETH difficulty) |
| $T_{total}$ | $T_1 + T_2 + T_H + T_{up} + T_{BC}$ | Linearithmic time |

## Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Funding

This work has been supported by the Kolarctic CBC project DIT4-BEARS under grant KO4096.

## References

[1] P. Thompson, A. James, A. Nanos, V-room: virtual meeting system trial, in: Proceedings of the 2013 IEEE 17th International Conference on Computer Supported Cooperative Work in Design (CSCWD); 27–29 Jun 2013; Whistler, BC, Canada, IEEE, Piscataway, NJ, USA, 2013, pp. 563–569.
[2] A.D. Bykov, V.I. Voronov, L.I. Voronova, et al., Web application development for biometric identification system based on neural network face recognition, in: 2020 Systems of Signals Generating and Processing in the Field of on Board Communications; 19–20 Mar 2020; Moscow, Russia, IEEE, Piscataway, NJ, USA, 2020, pp. 1–6.
[3] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system, Available online, http://bitcoin.org/bitcoin.pdf, 2008. (Accessed 17 January 2022).
[4] M. Alizadeh, K. Andersson, O. Schelén, A survey of secure Internet of things in relation to blockchain, J. Internet Serv. Inform. Secur. 3 (10) (2020) 47–75.
[5] M. Andoni, V. Robu, D. Flynn, et al., Blockchain technology in the energy sector: a systematic review of challenges and opportunities, Renew. Sustain. Energy Rev. 100 (2019) 143–174.
[6] J. Benet, IPFS, Available online, http://IPFS.io, 2014. (Accessed 17 January 2022).
[7] P. Lewis, T. Catanzano, L. Davis, S. Jordan, Web-based conferencing: what radiology educators need to know, Acad. Radiol. 27 (3) (2020) 447–454.
[8] S.L. Connolly, C.J. Miller, J.A. Lindsay, M.S. Bauer, A systematic review of providers' attitudes toward telemental health via videoconferencing, Clin. Psychol. Sci. Pract. 27 (2) (2020), e12311.
[9] S. Agrawal, A survey on recent applications of cloud computing in education: covid-19 perspective, J. Phys.: Conf. Ser. vol. 1828 (2021) 12076. IOP Publishing.
[10] K. Okereafor, P. Manny, Understanding cybersecurity challenges of telecommuting and video conferencing applications in the covid-19 pandemic, Int. J. IT Eng. 8 (6) (2020) 13–23.
[11] M. Mohanty, W. Yaqub, Seamless authentication for online teaching and meeting, in: 2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM); 24–26 Sep 2020; New Delhi, India, IEEE, Piscataway, NJ, USA, 2020, pp. 120–124.
[12] S. Dargan, M. Kumar, A comprehensive survey on the biometric recognition systems based on physiological and behavioral modalities, Expert Syst. Appl. 143 (2020) 113114.
[13] M. Elgharib, M. Mendiratta, J. Thies, M. Niessner, H.-P. Seidel, A. Tewari, V. Golyanik, C. Theobalt, Egocentric videoconferencing, ACM Trans. Graph. 39 (6) (2020) 1–16.
[14] M. Alizadeh, K. Andersson, O. Schelén, Efficient decentralized data storage based on public blockchain and IPFS, in: 2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE); 16–18 Dec 2020; Gold Coast, Australia, IEEE, Piscataway, NJ, USA, 2020, pp. 1–8.
[15] R. Kumar, P. Kumar, R. Tripathi, G.P. Gupta, T.R. Gadekallu, G. Srivastava, SP2F: a secured privacy-preserving framework for smart agricultural unmanned aerial vehicles, Comput. Network. 187 (2021) 107819.
[16] R. Kumar, R. Tripathi, N. Marchang, G. Srivastava, T.R. Gadekallu, N.N. Xiong, A secured distributed detection system based on IPFS and blockchain for industrial image and video data security, J. Parallel Distr. Comput. 152 (2021) 128–143.

[17] A. Mubashar, K. Asghar, A.R. Javed, et al., Storage and proximity management for centralized personal health records using an IPFS-based optimization algorithm, J. Circ. Syst. Comput. 31 (1) (2021), 2250010.
[18] S.S. Rao, A. Nahm, Z. Shi, X. Deng, A. Syamil, Artificial intelligence and expert systems applications in new product development—a survey, J. Intell. Manuf. 10 (3) (1999) 231–244.
[19] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, H. Balakrishnan, Chord: a scalable peer-to-peer lookup service for Internet applications, Comput. Commun. Rev. 31 (4) (2001) 149–160.
[20] P. Maymounkov, D. Mazieres, Kademlia: a peer-to-peer information system based on the XOR metric, in: P. Druschel, F. Kaashoek, A. Rowstron (Eds.), Peer-to-Peer Systems. IPTPS 2002, Springer, Heidelberg, Berlin, Germany, 2002, pp. 53–65.
[21] L.S. Sankar, M. Sindhu, M. Sethumadhavan, Survey of consensus protocols on blockchain applications, in: 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS); 6–7 Jan 2017; Coimbatore, India, IEEE, Piscataway, NJ, USA, 2017, pp. 1–5.
[22] J.B. Husić, S. Baraković, A. Veispahić, What factors influence the quality of experience for WebRTC video calls?, in: 2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO); 22–26 May 2017; Opatija, Croatia IEEE, Piscataway, NJ, USA, 2017, pp. 428–433.
[23] M. Adeyeye, I. Makitla, T. Fogwill, Determining the signalling overhead of two common WebRTC methods: JSON via XMLHttpRequest and SIP over WebSocket, in: 2013 Africon; 9–12 Sep 2013; Pointe aux Piments, Mauritius, IEEE, Piscataway, NJ, USA, 2013, pp. 1–5.
[24] E. André, N. Le Breton, A. Lemesle, et al., Comparative study of WebRTC open source SFUs for video conferencing, in: 2018 Principles, Systems and Applications of IP Telecommunications (IPTComm); 16–18 Oct 2018; Chicago, IL, USA, IEEE, Piscataway, NJ, USA, 2018, pp. 1–8.
[25] C. Li, C. Li, Web front-end realtime face recognition based on TFJS, in: 2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI); 19–21 Oct 2019; Suzhou, China, IEEE, Piscataway, NJ, USA, 2019, pp. 1–5.
[26] H. Klym, I. Vasylchyshyn, Face detection using an implementation running in a web browser, in: 2020 IEEE 21st International Conference on Computational Problems of Electrical Engineering (CPEE); 16–19 Sep 2020; Online Conference, IEEE, Piscataway, NJ, USA, 2020, pp. 1–4.
[27] face-api.js, Available online, https://justadudewhohacks.github.io/face-api.js/docs/index.html. (Accessed 17 January 2022).
[28] F. Filipovic, M. Despotovic-Zrakic, B. Radenkovic, et al., An application of artificial intelligence for detecting emotions in neuromarketing, in: 2019 International Conference on Artificial Intelligence: Applications and Innovations (IC-AIAI); 30 Sep–4 Oct 2019; Belgrade, Serbia, IEEE, Piscataway, NJ, USA, 2019, pp. 49–53.
[29] Orbit-db, Available online, https://github.com/orbitdb/orbit-db. (Accessed 17 January 2022).