SOMAIYA
VIDYAVIHAR
K J Somaiya Institute of Technology
An Autonomous Institute Permanently Affiliated to the University of Mumbai

DEPARTMENT OF INFORMATION TECHNOLOGY

Report On

TrustChain EMS using IPFS

Prepared By:

Ayush Samant (25)
Subhadip Samanta (26)
Niyati Sawant (28)

Under the guidance of:

Dr. Vijaya Umesh Pinjarkar

Department of Information Technology

Academic Year: 2024-2025

Autonomy Syllabus Scheme-II - SEMESTER VIII (LY - IT)

SOMAIYA
VIDYAVIHAR
K J Somaiya Institute of Technology
An Autonomous Institute Permanently Affiliated to the University of Mumbai

CERTIFICATE

This is to certify that following students:

	Roll No. / Seat No.
Ayush Samant	25
Subhadip Samanta	26
Niyati Sawant	28

have submitted Blockchain Mini-Project Report on *"**TrustChain EMS using IPFS**"* as the partial fulfillment for the requirement of Last Year of Engineering (8th Semester) in L.Y. - Information Technology under my guidance during the academic year 2023-2024.

Dr Vijaya Umesh Pinjarkar	**Dr. Radhika Kotecha**
Project Guide	**Head of Department**
Assistant Professor	**Professor**
Department of Information Technology	**Department of Information Technology**

Date of Examination: _____

Signature of Internal Examiner **Signature of External Examiner**

Table of Contents

Acknowledgement

Before presenting our mini-project entitled "**TrustChain EMS using IPFS**" we would like to convey our sincere thanks to the many people who guided us throughout the course for this project work. First, we would like to express our sincere thanks to our beloved Principal **Dr. Vivek Sunnapwar** for providing various facilities to carry out this project.

We would like to express our sincere thanks to **Dr. Vijaya Umesh Pinjarkar** for her guidance, encouragement, co-operation and suggestions given to us at progressing stages of the project. Finally, we would like to thank our H.O.D **Dr. Radhika Kotecha** and all teaching, non-teaching staff of the college and friends for their moral support rendered during the course of the project work and for their direct and indirect involvement in the completion of our project work, which made our endeavor fruitful.

Abstract

In the digital age, where the authenticity, integrity, and accessibility of evidence play a pivotal role in ensuring justice and transparency, traditional centralized evidence management systems are increasingly becoming vulnerable. Issues such as data tampering, unauthorized access, and the existence of single points of failure create significant risks that can undermine the credibility of the judicial process. To address these challenges, this project proposes the design and implementation of a TrustChain Evidence Management System (EMS) that integrates the InterPlanetary File System (IPFS) with blockchain technology.

The proposed solution utilizes blockchain's inherent characteristics of immutability, transparency, and distributed consensus mechanisms to create a decentralized and secure framework for managing digital evidence. By ensuring that evidence cannot be altered once recorded, the system provides a robust foundation for maintaining the integrity of crucial data, while also enabling its traceability and verifiability across time. Furthermore, the incorporation of smart contracts automates critical aspects of the system, such as access control, the maintenance of chain-of-custody records, and the enforcement of strict evidence handling policies, all without the need for a centralized authority.

This report provides a comprehensive outline of the TrustChain EMS architecture, its implementation details, the security protocols employed to protect sensitive data, and its potential applications within the legal and forensic domains, offering a modern solution to the evolving needs of evidence management in the digital era.

1. Introduction

In an era where digital evidence plays a pivotal role in law enforcement, judicial proceedings, and data governance, ensuring the authenticity, security, and availability of such evidence is paramount. Centralized systems often suffer from single points of failure, data tampering risks, and lack of transparency. This project integrates blockchain and IPFS (InterPlanetary File System) to provide a secure, tamper-proof, and transparent platform for managing digital evidence. Users can upload, verify, and retrieve evidence with full assurance of data integrity and provenance, without relying on centralized intermediaries.

1.1 Motivation

Traditional evidence management systems are frequently susceptible to manipulation, unauthorized access, and data loss. These vulnerabilities can compromise justice and undermine public trust. With blockchain offering immutability and decentralized verification, and IPFS enabling distributed file storage, this project is motivated by the need to build a system that not only secures digital evidence but also ensures transparent access and verifiability. By decentralizing storage and control, the platform aims to eliminate the weaknesses of centralized solutions while improving accessibility and trust among stakeholders.

1.2 Aim and Objective

The primary aim of this project is to develop a decentralized application that facilitates secure evidence management using blockchain and IPFS. The specific objectives are:

- To allow users to **upload evidence** of any file type, along with metadata including evidence ID, name, and owner information.
- To store the evidence file on IPFS and record its IPFS hash on the blockchain for integrity and traceability.
- To enable users to **download the evidence** securely using its IPFS hash.
- To provide a **verification function**, where a user inputs the evidence ID and retrieves details such as the evidence name, owner, file location, and IPFS hash.
- To maintain a tamper-proof record of all evidence entries and interactions on the blockchain using smart contracts.

1.3 Scope

This project encompasses the design and implementation of a decentralized evidence archive prototype. The system supports uploading of any digital file type, secure IPFS-based storage, blockchain-based metadata recording, and evidence verification via smart contracts. The platform ensures that only validated metadata is stored on the blockchain, while the actual files are distributed through IPFS. Although the current focus is on legal and forensic use cases, the framework is flexible and can be extended to other domains requiring secure digital asset management, such as academic certificates, financial records, or medical documentation.

2. Literature Review

In the paper *"Blockchain-based Decentralized Evidence Archive System using IPFS"*, Maharshi Dave and Dr. Rajkumar Banoth propose a decentralized platform aimed at enhancing the integrity and security of digital evidence in judicial processes. The authors address limitations of traditional centralized storage systems—such as data tampering, loss, and opacity—which often undermine legal proceedings [1]. By integrating the Ethereum blockchain with the InterPlanetary File System (IPFS), they develop a tamper-resistant and traceable storage architecture. The prototype allows users to upload digital evidence (text or image files), which is stored on IPFS, while the content hash is immutably recorded on the Ethereum blockchain via smart contracts written in Solidity. The system is implemented using Metamask, Remix IDE, and the Ropsten test network, with a web-based frontend to handle file submissions and retrieval [1][2][3].

A similar approach is discussed in the work by Luo et al., who design a **secure file sharing system** based on blockchain and IPFS [2]. Their system encrypts data before uploading to IPFS, and records the associated metadata on a consortium blockchain. Notably, this model supports role-based access control (RBAC), enhancing the security and confidentiality of evidence. Luo et al. highlight the latency and scalability challenges when using IPFS for large-scale file handling, suggesting future improvements through decentralized identity management and sharding techniques.

Expanding on these ideas, Waghmare et al. present an **evidence management system using blockchain and distributed file storage** [3]. Their focus lies on ensuring evidence authenticity throughout its lifecycle—from collection to presentation in court. They highlight how blockchain facilitates a tamper-evident audit trail, ensuring that any unauthorized alterations are instantly detectable. Their system provides timestamping, origin verification, and digital signatures for each uploaded file.

Furthering the forensic aspect, Bandara et al. propose a **chain-of-custody management system for digital forensic investigations** using blockchain [5]. Their work emphasizes legal admissibility of evidence, by ensuring each transition in the custody of digital evidence is recorded immutably. The proposed system is particularly relevant in criminal investigations where maintaining a robust, verifiable trail of digital artifacts is critical. Through smart contracts, they automate access permissions and enforce compliance to forensic standards.

Shilpa and Shanthakumara's 2023 study builds on previous designs by implementing a **crime evidence management system using blockchain and IPFS**, with a user-friendly interface that simplifies the process of evidence uploading and verification [6]. Their solution stresses user privacy by incorporating encryption techniques alongside IPFS, reducing risks of unauthorized access. The study also provides empirical results that support the feasibility of using these technologies even in resource-constrained environments.

Another relevant study by Sharma et al. discusses **integrating blockchain with IPFS for secure transfer and storage of digital evidence** [12]. They highlight a hybrid architecture that improves performance using a distributed peer-to-peer network for file handling, while ensuring long-term traceability via smart contracts on the blockchain. Their system includes modules for evidence classification, prioritization, and audit logging, presenting a more holistic framework suitable for law enforcement agencies.

Lastly, Loffi et al. conduct a **systematic literature review** on the management of digital evidence using blockchain and self-sovereign identities (SSI) [7]. They analyze over 60 research contributions, categorizing them into different application domains such as law enforcement, enterprise security, and civil litigation. Their review underscores the importance of interoperability, privacy-preserving mechanisms, and legal compliance, which are still underdeveloped in current systems.

3. Proposed System

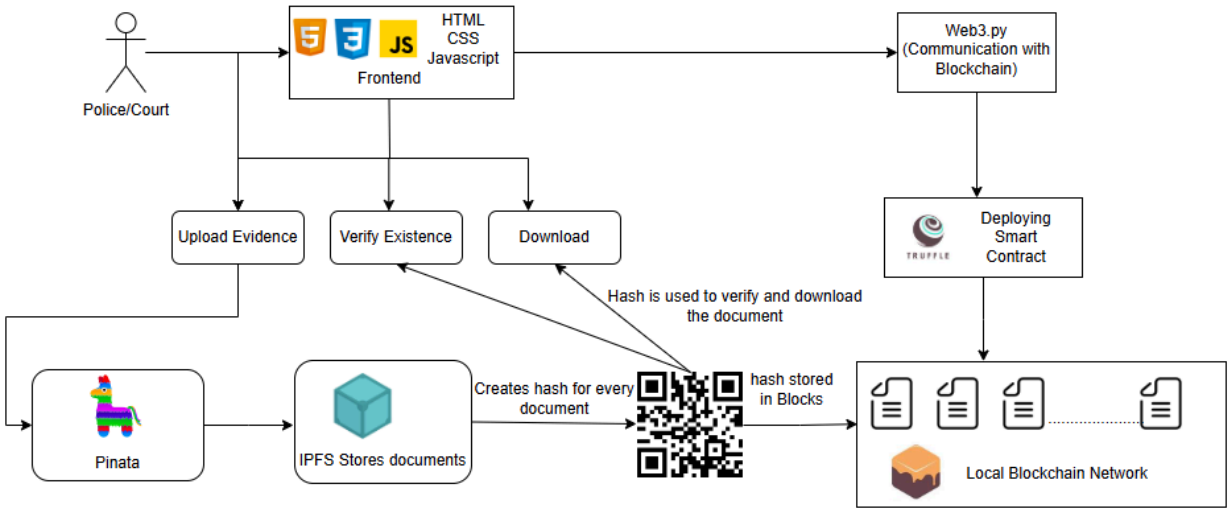

Fig 1: System Architecture

The proposed system is a **decentralized digital evidence management platform** built on a combination of blockchain and IPFS technologies. The system enables users to upload, verify, and download digital evidence through a user-friendly frontend, with secure backend processing powered by smart contracts and distributed storage. At the core, the system leverages:

- **HTML/CSS and JavaScript** for building a simple, user-friendly frontend interface for Police and Court users.
- **IPFS (via Pinata)** for decentralized storage of evidence files, ensuring permanent and tamper-proof storage.
- **Ethereum-based local blockchain network (Ganache)** to securely store the hash of each uploaded document, ensuring immutability and integrity.
- **Web3.js** for seamless communication between the frontend and the blockchain network.
- **Truffle framework** to compile, deploy, and manage smart contracts that handle evidence metadata securely.

3.2 Components Discussion

1. **Actor**: The end-users are Police Officers and Court Officials.
    - Police Officers can **upload** new digital evidence (audio, video, documents).

- Court Officials can **verify** existence and **download** evidence files.

2. **Frontend (HTML/CSS/JavaScript)**: A simple web interface where users can **Register/Login** by providing:
   - Name
   - Role (Police or Court)
   - Password

   After logging in, depending on the role:
   - Police can upload evidence with metadata (Location, Crime Description, etc.).
   - Court can verify the existence of evidence and download it.
3. **IPFS Storage via Pinata**: Files uploaded by users are pinned to IPFS using Pinata. Each file generates a unique IPFS hash, ensuring decentralized and tamper-proof storage.
4. **Hash Generator**: After uploading, each file's hash can be optionally converted into a hash for easier verification and access.
5. **Smart Contract (Truffle + Solidity)**: Smart contracts record and secure:
   - Evidence metadata (e.g., Evidence ID, uploader's details, location, description)
   - IPFS file hash - Stored on the local Ethereum blockchain (Ganache), this ensures evidence is immutable.
6. **Web3.js**: **Web3.js** connects the frontend to the blockchain, enabling users to:
   - Send transactions
   - Retrieve or verify stored evidence information
7. **Local Blockchain Network (Ganache)**: A simulated Ethereum blockchain is used during development to:
   - Deploy smart contracts
   - Test blockchain interactions
   - Simulate real-world blockchain behavior without incurring gas fees

3.3 Workflow

Upload Evidence (Police Role):

1. Police Officer logs in using Name, Role, and Password.
2. Fills a form to upload:
    - Evidence file (audio, video, document, etc.)
    - Location of crime
    - Crime description
    - Additional metadata if needed.
3. File is uploaded to **Pinata (IPFS)**; a **content-addressed hash** is generated.
4. The frontend interacts with the smart contract (via Web3.js) to store:
    - Evidence ID
    - Uploader details
    - IPFS hash
    - Location
    - Crime description

Verify Evidence Existence (Court Role):

1. Court Official logs in using Name and Role
2. Inputs the Evidence ID to verify.
3. The system queries the smart contract to fetch:
    - IPFS hash
    - Uploader information
    - Metadata (Location, Crime Description)
4. Verification confirms the existence and authenticity of the evidence.

Download Evidence (Court Role):

1. Once verified, the Court Official can retrieve the IPFS hash.
2. Using the IPFS hash, the evidence file is downloaded directly from IPFS (via Pinata).

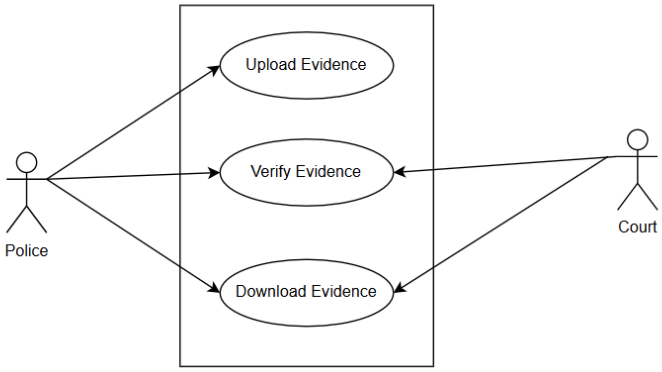

Fig 2 Use Case Diagram

Figure 2 represents the use case diagram . The police can upload , verify and download the evidence while the court can only verify and download it.

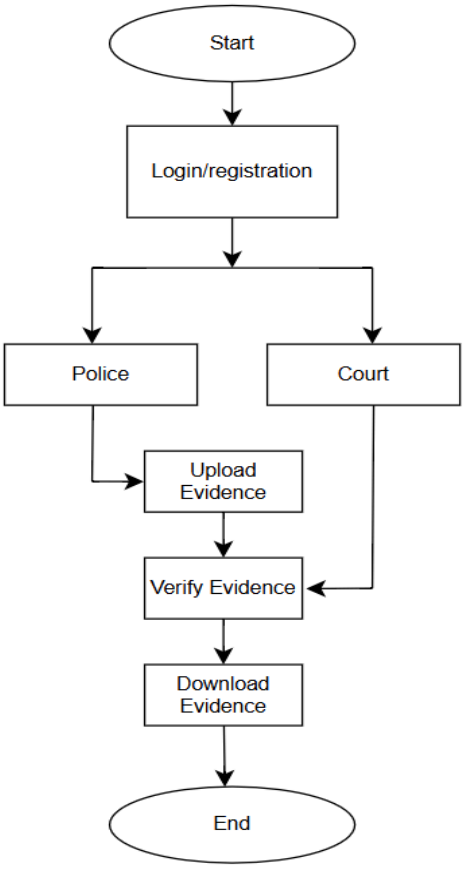

Fig 3 Workflow Diagram

Figure 3 represents the workflow diagram showing user registration, police uploading/verifying/downloading evidence, and court verifying/downloading evidence in the blockchain-based system

4 Implementation Details

4.1 Tech Stack

The system is built using a modern decentralized technology stack to ensure trust, transparency, and data immutability. The main components of the tech stack include:

- **Frontend:**
    - **HTML/CSS and JavaScript** – Basic web technologies are used to create a user-friendly frontend interface for uploading, verifying, and downloading evidence.
    - **Web3.js** – A JavaScript library that enables communication between the frontend and the Ethereum blockchain.
    - **MetaMask** – A browser extension that acts as a blockchain wallet, allowing users to sign transactions and interact securely with the blockchain network.
- **Blockchain Layer:**
    - **Ethereum (Local Test Network)** – A local blockchain environment (such as Ganache) is used to simulate smart contract deployment and interactions.
    - **Solidity** – The programming language used to write smart contracts that store and manage evidence metadata securely on the blockchain.
    - **Truffle Suite** – A development framework for Ethereum used to compile, migrate, and test smart contracts.
    - **Web3.py** – A Python library that facilitates communication between the Streamlit frontend and the Ethereum blockchain.
- **Storage Layer:**
    - **IPFS (InterPlanetary File System)** – A peer-to-peer distributed file system used to store the actual evidence files. It returns a unique content-addressable hash for every file uploaded.
    - **Pinata** – An IPFS pinning service used to ensure that uploaded files remain persistently available on IPFS.

- **Tools & Environment:**
    - **MetaMask** – For simulating blockchain wallet functionality and account-based interactions.
    - **Ganache** – To simulate a local Ethereum blockchain for testing smart contracts.
    - **Python 3.10** – The primary programming language used across the system.

4.2 Database Description

In this decentralized system, traditional databases are replaced by a combination of IPFS and blockchain to manage and store information securely:

- **IPFS Storage:**
    - Acts as the decentralized file storage system for storing evidence files.
    - Each uploaded file is stored via Pinata on IPFS and returns a unique content-addressable hash.
    - The files can be accessed from any IPFS gateway using the hash, ensuring fault tolerance and decentralization.
- **Blockchain Ledger:**
    - Stores metadata related to each piece of evidence such as:
        - Evidence ID (as the primary key)
        - Evidence Name
        - Owner Name
        - Crime Location
        - Crime Description
        - IPFS Hash (for file retrieval)
    - This ledger is immutable and ensures the integrity and authenticity of the evidence.
    - The smart contract serves as the logic layer that ensures controlled access and verifiability of the evidence records.

4.3 Screenshot of project including Smart Contract Code

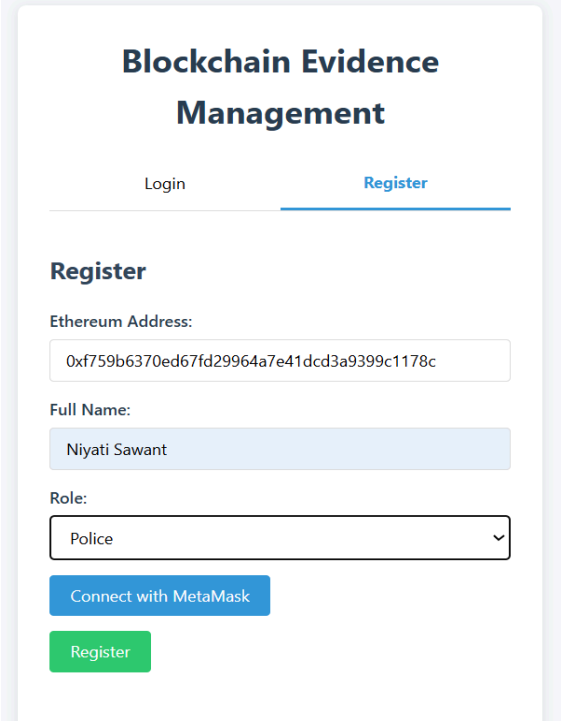

Fig 1 : Registration page

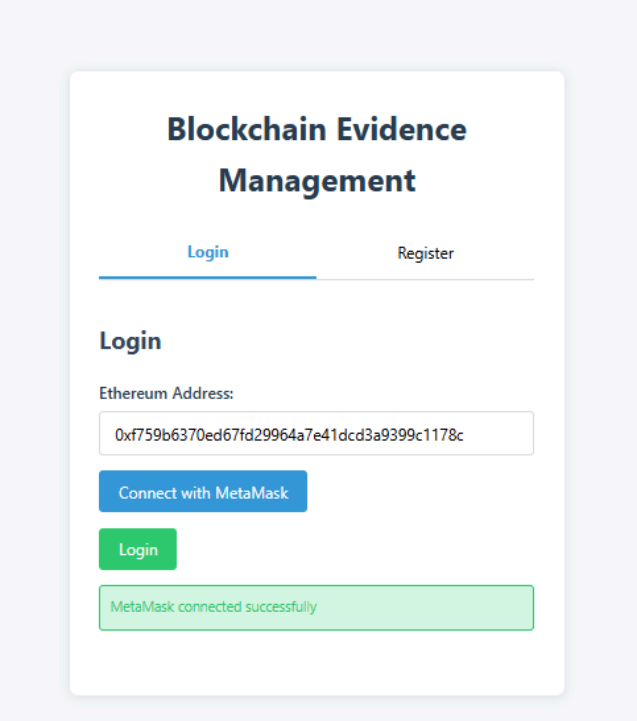

Fig 2 : Login page

Figure 1 shows the Registration page depending on the role Police/Court while Fig 2 depicts the login page

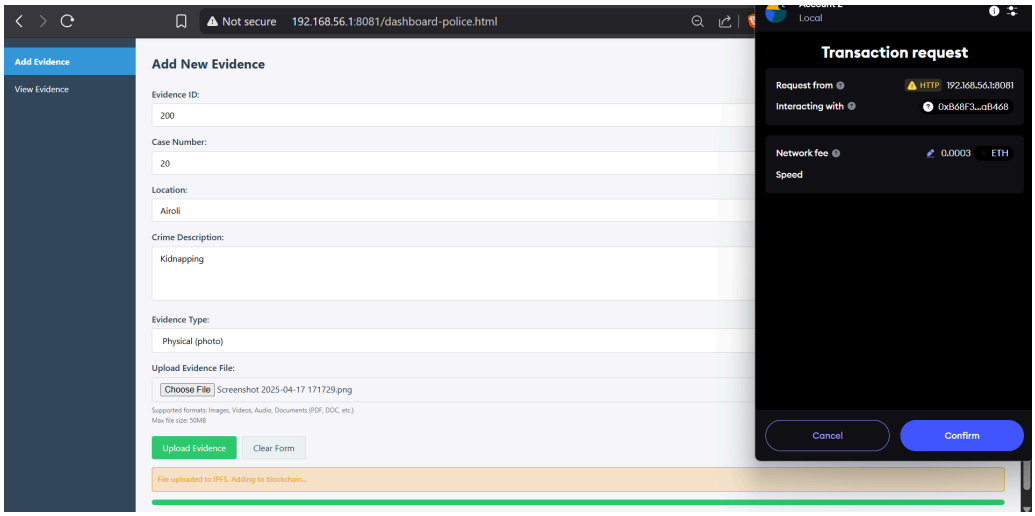

Fig 2 :Upload Evidence by Police

Figure 2 shows the "Upload Evidence" page designed for Police users. It allows police officers to enter evidence details, upload files (audio, video, documents, etc.), and submit them to the decentralized storage system.

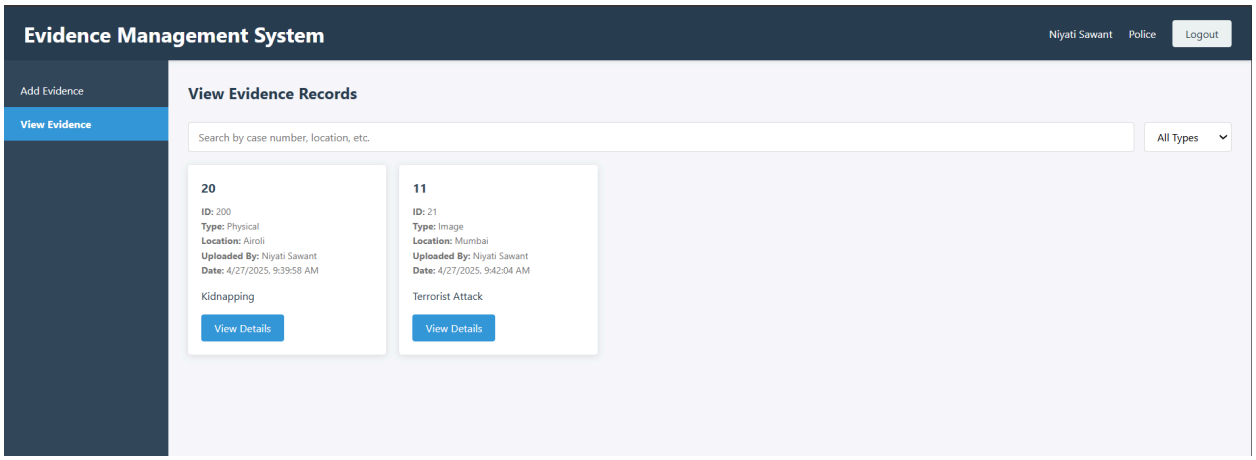

Fig 3 :Verify Evidence Dashboard for Police

Figure 3 displays the "Verify Evidence" page for Police. It enables officers to verify the existence and integrity of previously uploaded evidence using evidence metadata and IPFS hashes.

Fig 4: View and Download Evidence

Figure 4 presents the "View and Download Evidence" page for Investigators. It allows investigators to search, view metadata, and securely download evidence files from IPFS.

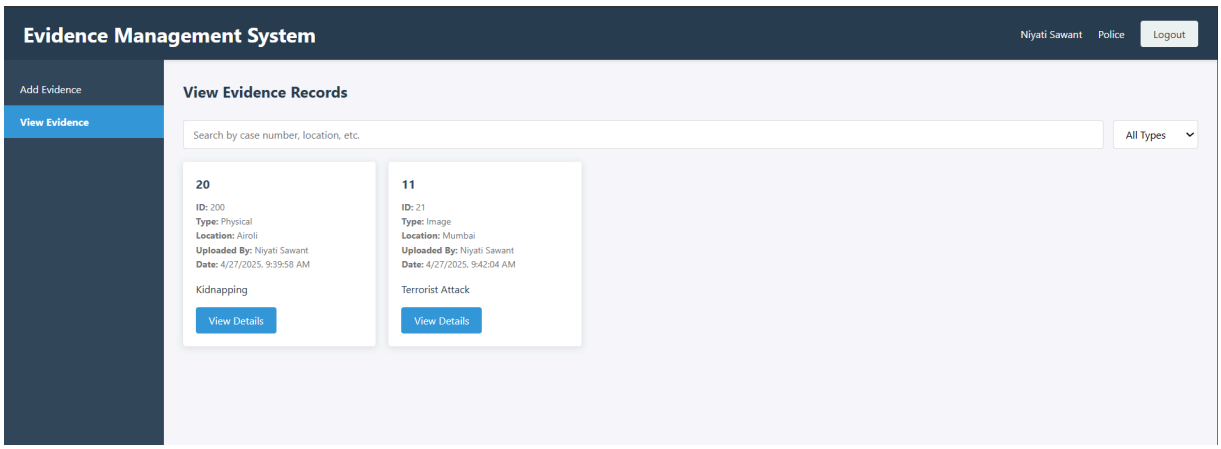

Fig 5 :View Evidence Dashboard for Police

Figure 5 shows the "View Evidence" dashboard for Police officers. It provides an overview of all uploaded evidence entries, including their metadata and access to detailed records.

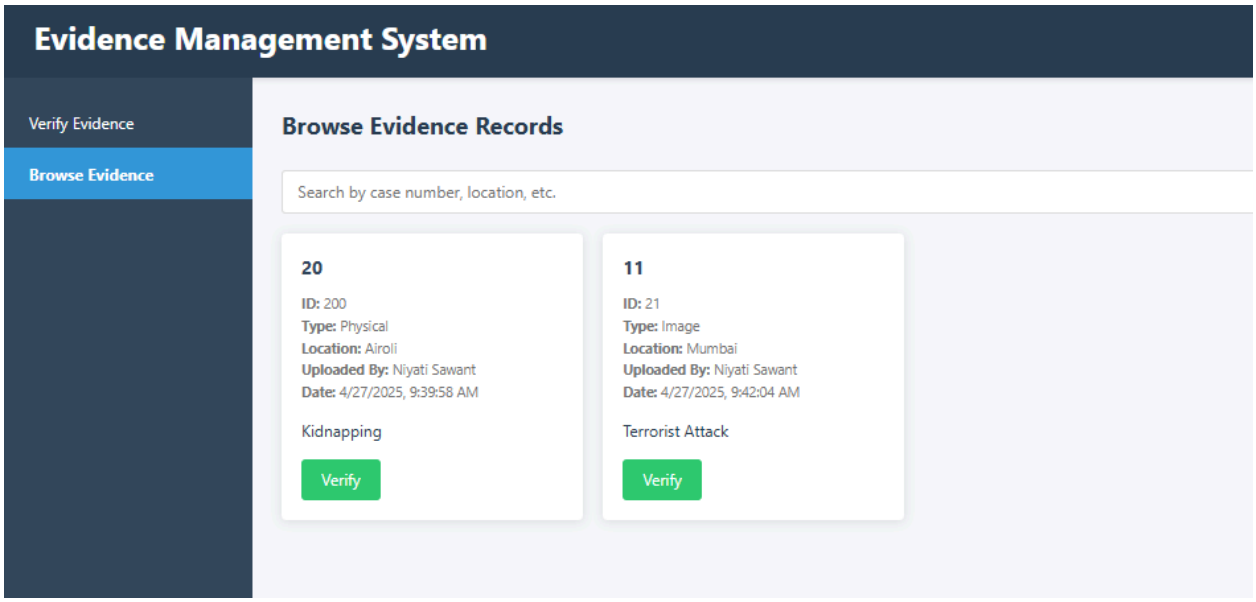

Fig 6 :Browse Evidence for court

Figure 6 illustrates the "Browse Evidence" page for Court officials. Courts can browse all submitted evidence, view related details, and prepare for judicial verification.

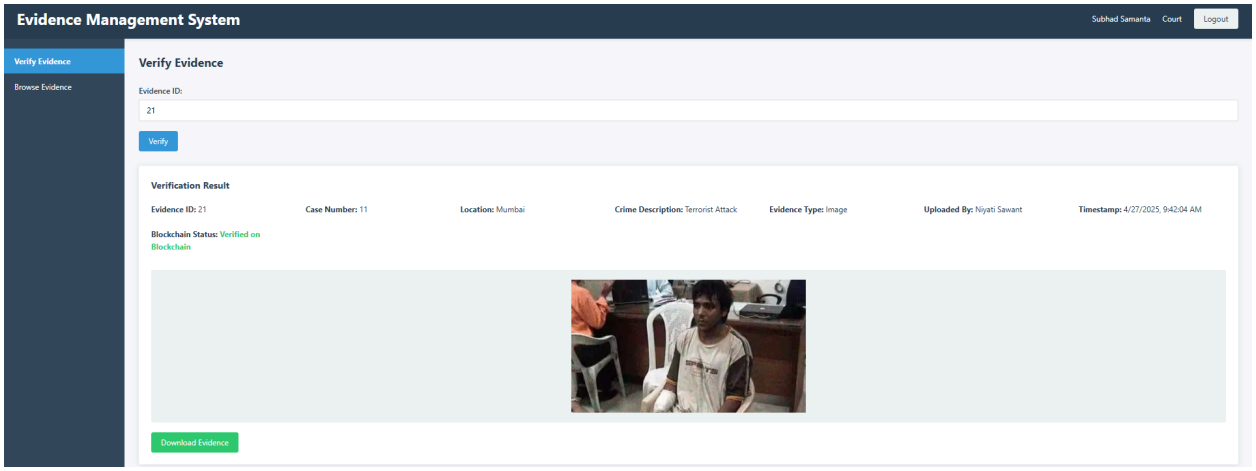

Fig 7 : Verify Evidence Page for Court

Figure 7 depicts the "Verify Evidence" page for Court officials. It facilitates the verification process by allowing courts to confirm the authenticity and availability of submitted evidence through blockchain records.

Smart Contract Code

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
contract EvidenceManagement {
    // Struct for evidence details
    struct Evidence {
        string ipfsHash;        // IPFS hash of the evidence file
        string caseNumber;      // Unique case identifier
        string location;        // Location where evidence was
collected
        string crimeDescription; // Brief description of the crime
        string evidenceType;    // Type of evidence (audio, video,
document, etc.)
        string officerName;     // Name of the officer who uploaded
the evidence
        uint256 timestamp;      // Timestamp when evidence was added
        bool exists;            // Flag to check if evidence exists
    }
    // Mapping from evidence ID to Evidence struct
    mapping(string => Evidence) public evidenceRecords;
    // List of all evidence IDs for iteration
    string[] public evidenceIds;
    // Authorized users mapping (address => role)
    mapping(address => string) public userRoles;
    mapping(address => string) public userNames;
    mapping(address => bool) public registeredUsers;
    // Events
    event EvidenceAdded(string evidenceId, string ipfsHash, string
officerName, uint256 timestamp);
    event EvidenceAccessed(string evidenceId, address accessedBy,
uint256 timestamp);
    event UserRegistered(address userAddress, string role, string
name);
    // Modifiers
    modifier onlyPolice() {
        require(
            keccak256(abi.encodePacked(userRoles[msg.sender])) ==
keccak256(abi.encodePacked("Police")),
```

```
            "Only police officers can add evidence"
        );
        _;
    }
    modifier onlyRegistered() {
        require(registeredUsers[msg.sender], "User not registered");
        _;
    }
    // Register a new user
    function registerUser(string memory _name, string memory _role)
public {
        require(!registeredUsers[msg.sender], "User already
registered");
        require(
            keccak256(abi.encodePacked(_role)) ==
keccak256(abi.encodePacked("Police")) ||
            keccak256(abi.encodePacked(_role)) ==
keccak256(abi.encodePacked("Court")),
            "Role must be either Police or Court"
        );
        userRoles[msg.sender] = _role;
        userNames[msg.sender] = _name;
        registeredUsers[msg.sender] = true;
        emit UserRegistered(msg.sender, _role, _name);
    }
    // Add new evidence (only police)
    function addEvidence(
        string memory _evidenceId,
        string memory _ipfsHash,
        string memory _caseNumber,
        string memory _location,
        string memory _crimeDescription,
        string memory _evidenceType
    ) public onlyPolice onlyRegistered {
        require(!evidenceRecords[_evidenceId].exists, "Evidence ID
already exists");
        Evidence memory newEvidence = Evidence({
            ipfsHash: _ipfsHash,
            caseNumber: _caseNumber,
            location: _location,
```

```
            crimeDescription: _crimeDescription,
            evidenceType: _evidenceType,
            officerName: userNames[msg.sender],
            timestamp: block.timestamp,
            exists: true
        });
        evidenceRecords[_evidenceId] = newEvidence;
        evidenceIds.push(_evidenceId);
        emit EvidenceAdded(_evidenceId, _ipfsHash,
userNames[msg.sender], block.timestamp);
    }
    // Get evidence details
    function getEvidence(string memory _evidenceId) public
onlyRegistered returns (
        string memory ipfsHash,
        string memory caseNumber,
        string memory location,
        string memory crimeDescription,
        string memory evidenceType,
        string memory officerName,
        uint256 timestamp
    ) {
        require(evidenceRecords[_evidenceId].exists, "Evidence does
not exist");
        Evidence memory evidence = evidenceRecords[_evidenceId];
        emit EvidenceAccessed(_evidenceId, msg.sender,
block.timestamp);
        return (
            evidence.ipfsHash,
            evidence.caseNumber,
            evidence.location,
            evidence.crimeDescription,
            evidence.evidenceType,
            evidence.officerName,
            evidence.timestamp
        );
    }
     // Get all evidence IDs
    function getEvidenceCount() public view returns (uint256) {
        return evidenceIds.length;
```

```
    }
     // Check if user is registered
    function isUserRegistered(address _userAddress) public view
returns (bool) {
        return registeredUsers[_userAddress];
    }
     // Get user role
    function getUserRole(address _userAddress) public view returns
(string memory) {
        require(registeredUsers[_userAddress], "User not
registered");
        return userRoles[_userAddress];
    }
    // Get user name
    function getUserName(address _userAddress) public view returns
(string memory) {
        require(registeredUsers[_userAddress], "User not
registered");
        return userNames[_userAddress];
    }
}
```

5. Results & Discussion

The developed system was tested successfully on a local blockchain network integrated with IPFS for decentralized file storage. The primary goal of ensuring tamper-proof, verifiable digital evidence management was achieved through the seamless interaction between smart contracts, IPFS, and the user-facing frontend.

Key Results:

1. **Evidence Upload Functionality:**
    - Users were able to upload any type of file (PDF, image, video, etc.) along with associated metadata like Evidence ID, Name, and Owner.
    - Files were securely stored on IPFS, and a unique content-addressed hash was generated for each.
    - This hash was recorded immutably on the blockchain via a deployed smart contract.
2. **Verification Feature:**
    - By entering an Evidence ID, users could query the smart contract and retrieve:
        - Evidence Name
        - Owner Name
        - File Location (IPFS hash)
    - This confirmed the existence and integrity of the uploaded file, proving resistance to tampering.
3. **File Download:**
    - Users could download evidence using its IPFS hash.
    - This demonstrated the decentralized and distributed nature of storage—files remained accessible without relying on a single centralized server.

Limitations:

- The current implementation uses a **local blockchain** for testing purposes. For real-world deployment, a public or permissioned blockchain would be needed.
- **Privacy and access control** are basic in this version. Any user with an IPFS hash can download the file, which could be improved using encryption or authentication layers.

- **File size constraints** may affect IPFS performance depending on the network setup and node availability.

5.1 Observations

- Evidence Authenticity: The system successfully prevents evidence tampering and duplication by ensuring each evidence record is immutably stored and associated with a unique identifier on the blockchain.
- User Experience: The React-based frontend (or DApp interface), combined with MetaMask wallet integration for user verification, provided a smooth and intuitive experience for police officers, court officials, and administrators.
- Decentralization: With all critical evidence metadata and user authentication handled through smart contracts, the platform operates without any centralized server or database, ensuring true decentralization and trustless operations.
- Transaction Time: While evidence uploads and access logs were secure and immutable, they depended on Ethereum network conditions and gas fees, which could occasionally affect responsiveness during peak network usage.

5.2 Vulnerability Analysis using Slither and Mythril

1. **Analysis using Slither-**Slither, a powerful static analysis tool for Solidity smart contracts, was utilized to detect vulnerabilities, optimization opportunities, and adherence to best coding practices. After running Slither on the smart contract, the following observations were made:
    - Assembly Usage: Instances of inline assembly were detected in imported OpenZeppelin utility contracts. Although assembly can optimize gas usage and performance, it demands cautious use to avoid security risks. In this project, assembly was handled appropriately.
    - Compiler Version Alignment: Minor inconsistencies in Solidity compiler versions across imported contracts were observed. Aligning compiler versions is recommended to ensure maximum compatibility and minimize unforeseen behavior.
    - Low-Level Call Handling: The contract uses low-level functions such as call and sendValue. Proper success verification checks were incorporated to handle these operations securely and prevent vulnerabilities like reentrancy.

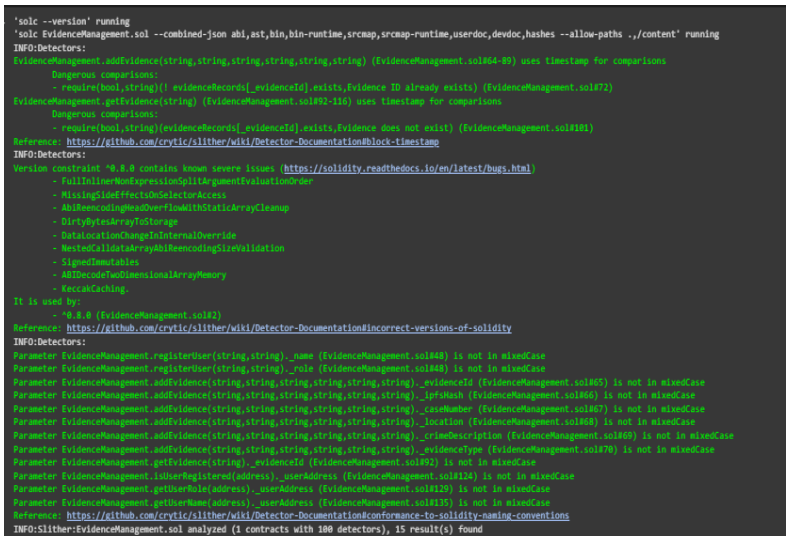

Fig 5.2.1- Implementation of Slither

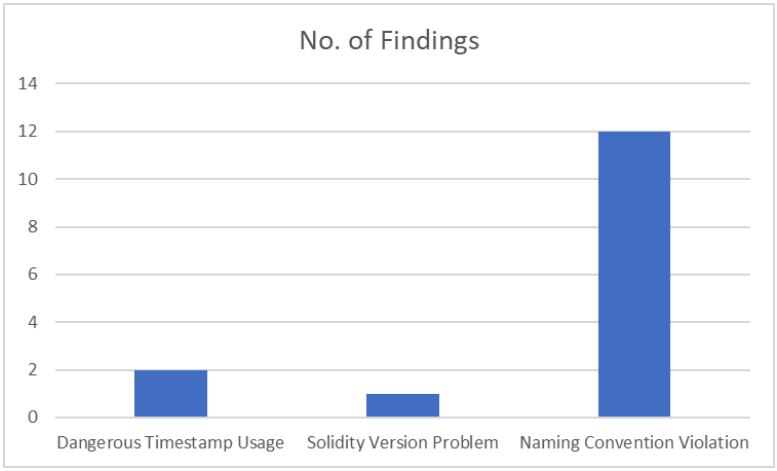

Fig 5.2.2- Histogram of findings

2. Analysis using Mythril - Mythril module focuses specifically on detecting integer-related vulnerabilities like overflows and underflows. The following observations were made after executing analysis:

- Integer Overflow/Underflow Detection: No critical integer overflow or underflow vulnerabilities were identified in the smart contract's arithmetic operations. This indicates that the contract effectively handles numeric computations without risking security errors.
- Arithmetic Safety: Operations involving critical numerical values were found to be safe, either due to Solidity's built-in overflow checks (introduced from Solidity version 0.8.0 onwards) or due to secure implementation practices.

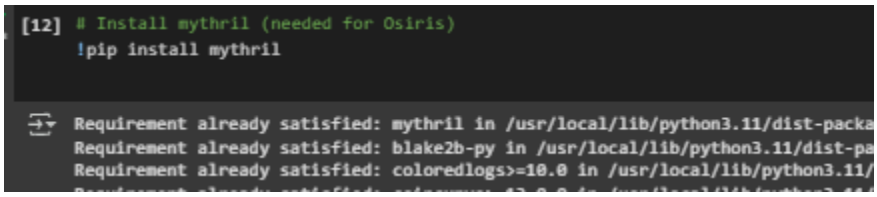

Fig 5.2.3 - Installation of mythril

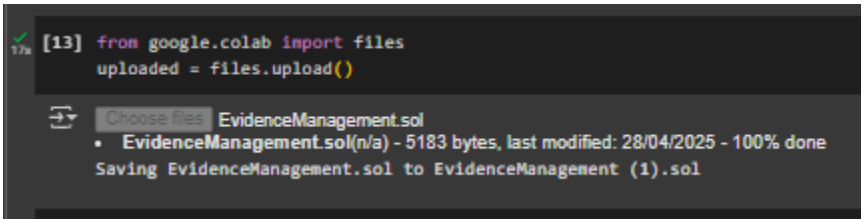

Fig 5.2.3- Uploading Smart contract

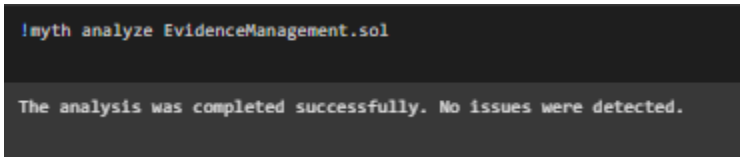

Fig 5.2.4 - Results

Vulnerability Type	**Detected**
Reentrancy	No
Integer Overflow	No
Gas Limit Issues	No
Uninitialized Variables	No
Unprotected Functions	No

Table 1 - Mythril Vulnerability Analysis

Vulnerability Type	Detected by Slither	Detected by Mythril
Reentrancy	No	No
Integer Overflow	No	No
Gas Limit Issues	No	No
Uninitialized Variables	No	No
Unprotected Functions	No	No
Dangerous Timestamp Comparison	Yes	No
Naming Convention Issues	Yes	No

Comparison of Vulnerability Detection Between Slither and Mythril

5.3 Comparison of proposed system with Existing system

Feature	**Existing Centralized Evidence Systems**	**TrustChain EMS**
Platform Type	Centralized servers and databases	Decentralized blockchain-based system
Evidence Storage	Stored in private/internal databases	IPFS for files + Metadata stored on blockchain
Evidence Ownership Verification	Controlled by database admin, manual verification	Blockchain-backed automatic verification (tamper-proof)
Fraud Prevention	Moderate – depends on internal controls and audits	Strong – cryptographic validation, immutable records
Access Control	Login-based, admin approval needed	Role-based on-chain (Police/Court), wallet address verification
Data Transparency	Low – internal records hidden from public view	High – blockchain data is publicly verifiable (audit trail)
Intermediary Trust	Full trust in system admins and officers	Trustless system – blockchain enforces rules automatically
Intermediary Fees	High – maintenance, server costs, audit costs	Low – only minimal gas fees for transactions
User Privacy	Requires full personal data, logs stored centrally	Wallet address-based, minimal personal information required
Scalability	Limited to internal networks and manual expansions	Scalable globally – accessible through any blockchain frontend
Tamper Evidence	Hard to detect subtle data tampering internally	Instantly detectable – hash mismatches or unauthorized changes impossible

6. Conclusion

The Blockchain-Based Decentralized Evidence Archive System successfully demonstrates the potential of integrating blockchain and IPFS technologies to build a secure, transparent, and tamper-proof digital evidence management platform. By leveraging the immutability of smart contracts and the decentralized storage capabilities of IPFS, the system ensures that digital evidence remains verifiable, accessible, and resistant to unauthorized modifications. The user-friendly Streamlit interface further enhances accessibility, allowing even non-technical users to upload, verify, and retrieve evidence with ease. While the current implementation is limited to a local blockchain environment and lacks advanced access control features, the project lays a strong foundation for scalable real-world applications in legal, academic, and forensic domains. Future enhancements such as encryption, user authentication, and deployment on public or permissioned blockchains can further elevate the system's practicality and security.

7. Future Work

8. Future Scope

- **Enhanced Security with Encryption**:
  Integrate encryption mechanisms (e.g., AES or RSA) to protect the contents of uploaded evidence files, ensuring confidentiality even if the IPFS hash is exposed.

- **Deployment on Public/Consortium Blockchain**:
  Shift from local Ethereum testnet to public (e.g., Ethereum mainnet, Polygon) or consortium blockchains (e.g., Hyperledger) for real-world deployment and scalability.

- **Audit Logs & Activity Tracking**:
  Introduce detailed audit logs to track all interactions with the evidence data for compliance and forensic review.

- **AI Integration for Evidence Categorization**:
  Integrate machine learning to automatically tag, categorize, or flag potential duplicate or tampered evidence files.

- **IPFS Node Hosting on Cloud Services**: Host dedicated IPFS nodes on cloud platforms (e.g., AWS, GCP) to ensure high availability and faster file retrieval.

8. References

[1] Dave, Maharshi, and Rajkumar Banoth. "Blockchain-based, Decentralized Evidence Archive System using IPFS." 2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS). IEEE, 2022.

[2] Luo, Quan, et al. "A secure file sharing system based on IPFS and blockchain." *2022 IEEE International Conference on Computer Communication and the Internet (ICCCI)*. IEEE, 2022.

[3] Waghmare, Ramesh, et al. "Evidence Management System Using Blockchain and Distributed File System (IPFS)." *2021 International Journal of Scientific Research in Engineering and Management (IJSREM)*, vol. 5, no. 9, 2021.

[4] N, Satwik, et al. "Integrating Public Reported Evidence Collection and Public Court Records Archive Using IPFS and Hyperledger Fabric Blockchain." *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 11, no. 5, 2023.

[5]P. B. Bandara, O. D. Jayarathna, D. T. Hewage, N. P. Bandara, D. Pandithage and D. Siriwardana, "Blockchain-Based Chain of Custody Evidence Management System for Digital Forensic Investigations," 2023 5th International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 2023

[6] C. Shilpa and A. H. Shanthakumara, "An Implementation of Blockchain Technology in Combination with IPFS for Crime Evidence Management System," 2023 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2023

[7] L. Loffi, G. L. Camillo, C. A. D. Souza, C. M. Westphall and C. B. Westphall, "Management of the Chain of Custody of Digital Evidence Using Blockchain and Self-Sovereign Identities: A Systematic Literature Review," in IEEE Access

[8]K. M. Mahdi Salih and N. B. Ibrahim, "CustodyChainGuardian: Blockchain of Custody Digital Evidence Preservation System," 2023 IEEE Asia-Pacific Conference on Geoscience, Electronics and Remote Sensing Technology (AGERS), Surabaya, Indonesia, 2023

[9]A. Sonawane, N. Lakade, S. Sarkar and N. Dongre, "Securing Digital Evidence In Forensic Investigation Using Blockchain," 2024 4th Asian Conference on Innovation in Technology (ASIANCON), Pimari Chinchwad, India, 2024

[10] R. Sathyaprakasan, P. Govindan, S. Alvi, L. Sadath, S. Philip and N. Singh, "An Implementation of Blockchain Technology in Forensic Evidence Management," 2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), Dubai, United Arab Emirates, 2021

[11] S. K. Verma, A. Gupta, and S. K. Bansal, "Blockchain Technology for Secure and Transparent Digital Evidence Management," 2023 International Conference on Computational Intelligence and Data Science (ICCIDS), Hyderabad, India, 2023.

[12] M. Sharma, R. K. Sharma, and S. R. Laskar, "Integrating Blockchain with IPFS for Secure Storage and Transfer of Digital Evidence," 2022 International Conference on Cloud Computing and Data Science (ICCCDS), New Delhi, India, 2022.

[13] A. G. V. Bhandari and M. S. Gohil, "A Review of Blockchain-Based Digital Evidence Storage and Retrieval Systems," 2023 International Conference on Digital Forensics and Cybersecurity (ICDFCS), London, United Kingdom, 2023.

[14] K. P. Yadav and A. V. R. Reddy, "Blockchain-Driven Chain of Custody Model for Forensic Evidence Authentication," 2024 International Symposium on Computer Science and Engineering (ISCSE), Bangalore, India, 2024.

[15] M. B. Patel, R. A. Sharma, and A. R. Singh, "Blockchain-Based Digital Evidence Management Framework for Forensic Investigations," 2023 IEEE International Conference on Advanced Computing (ICAC), Boston, USA, 2023.