DrillBit

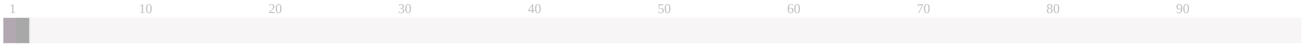
The Report is Generated by DrillBit Plagiarism Detection Software

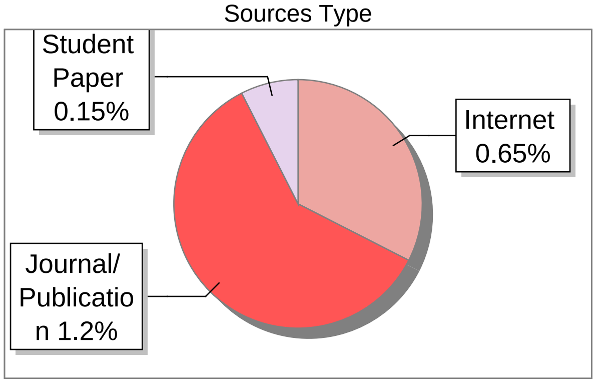
Submission Information

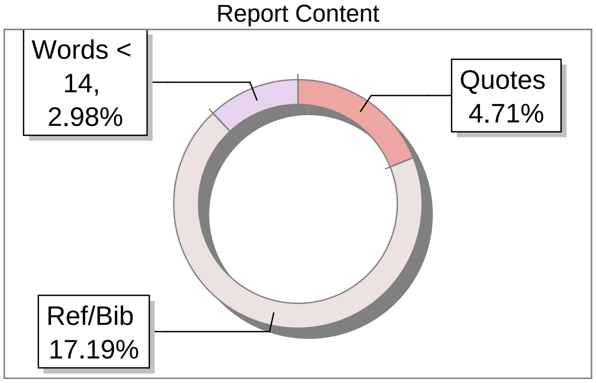
Author Name	Pratheek G Shetty 4SF22CS148
Title	Leveraging Blockchain for secure law enforcement
Paper/Submission ID	4770826
Submitted by	shwetha.library@sahyadri.edu.in
Submission Date	2025-11-29 17:00:52
Total Pages, Total Words	79, 19058
Document type	Project Work

Result Information

Similarity **2 %**




Sources Type


Report Content

Exclude Information

Quotes	Excluded
References/Bibliography	Excluded
Source: Excluded < 14 Words	Excluded
Excluded Source	**0 %**
Excluded Phrases	Not Excluded

Database Selection

Language	English
Student Papers	Yes
Journals & publishers	Yes
Internet or Web	Yes
Institution Repository	Yes

A Unique QR Code use to View/Download/Share Pdf File

DrillBit Similarity Report

2	8	A	A-Satisfactory (0-10%) B-Upgrade (11-40%) C-Poor (41-60%) D-Unacceptable (61-100%)
SIMILARITY %	MATCHED SOURCES	GRADE	

LOCATION	MATCHED DOMAIN	%	SOURCE TYPE
1	www.bmsce.ac.in	<1	Publication
2	moam.info	<1	Internet Data
3	ieeexplore.ieee.org	<1	Publication
4	information-science-engineering.newhorizoncollegeofengineering.in	<1	Publication
6	index-of.es	<1	Publication
7	www.nature.com	<1	Internet Data
18	REPOSITORY - Submitted to MVJ College of Engineering Bengaluru on 2025-11-26 15-07 4746934	<1	Student Paper
19	alameenedu.co.in	<1	Publication

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
“JNANA SANGAMA”, BELAGAVI - 590 018

PROJECT PHASE - II REPORT

on

“Leveraging Blockchain for secure law enforcement and government records.”

Submitted by

Himanshu S Shetty	4SF22CS078
M Imaad Iqbal	4SF22CS111
Pratheek G Shetty	4SF22CS148
Shifali Florine Lobo	4SF22CS192

In partial fulfillment of the requirements for the VII semester

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE & ENGINEERING

Under the Guidance of

Dr. Mustafa Basthikodi

HOD, Department of CSE

at

SAHYADRI

College of Engineering & Management

An Autonomous Institution

MANGALURU

2025 - 26

SAHYADRI
COLLEGE OF ENGINEERING & MANAGEMENT
An Autonomous Institution
MANGALURU

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the phase - II work of project entitled **“ Leveraging Blockchain for secure law enforcement and government records.”** has been carried out by **Himanshu S Shetty (4SF22CS078),M Imaad Iqbal (4SF22CS111), Pratheek G Shetty (4SF22CS148) and Shifali Florine Lobo (4SF22CS192)**, the bonafide students of Sahyadri College of Engineering and Management [19]in partial fulfillment of the requirements for the VII [4]semester of Bachelor of Engineering in [1]Computer Science and Engineering of Visvesvaraya Technological University, Belagavi during the year 2025 - 26. It is certified that all [4]suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the said degree.

Dr. Mustafa Basthikodi	**Dr. S S Injaganeri**
Professor & Head	Principal
Dept. of CSE	SCEM

Examiner’s Name	**Signature with Date**
1.	....
2.	....

SAHYADRI
COLLEGE OF ENGINEERING & MANAGEMENT
An Autonomous Institution
MANGALURU

Department of Computer Science & Engineering

DECLARATION

We hereby declare that the entire work embodied in this Project Phase - II Report titled **“Leveraging Blockchain for secure law enforcement and government records”** has been carried out by us at Sahyadri College of Engineering and Management, Mangaluru under the supervision of **Dr. Mustafa Basthikodi.,** in partial fulfillment of the requirements for the VII semester of **Bachelor of Engineering** in **Computer Science and Engineering**. This report has not been submitted to this or any other University for the award of any other degree.

Himanshu S Shetty	**(4SF22CS078)**
M Imaad Iqbal	**(4SF22CS111)**
Pratheek G Shetty	**(4SF22CS148)**
Shifali Florine Lobo	**(4SF22CS192)**

Dept. of CSE, SCEM, Mangaluru

ABSTRACT

The integrity, security, and accessibility of government and law enforcement records are fundamental to public trust. Yet, traditional centralized databases remain vulnerable to data manipulation, cyberattacks, and systemic inefficiencies that can compromise sensitive legal documents and undermine judicial processes. Blockchain technology offers a paradigm shift toward decentralized trust and cryptographic assurance. This paper presents a robust framework designed using Hyperledger Fabric to secure government legal records and enhance law enforcement procedures. We focus on the practical application of a permissioned blockchain, which is better suited for government use than public, cryptocurrency-based models. The architecture integrates smart contracts to automate legal documentation workflows and employs a hybrid on-chain/off-chain storage model to ensure scalability. Our results demonstrate a functionally complete, tamper-proof system that improves efficiency and provides a fully auditable trail for all record interactions. This work validates the feasibility of blockchain as a foundational technology for next-generation digital governance.

ACKNOWLEDGEMENT

It is with great satisfaction and euphoria that we are submitting the Project Phase - II Report on **“Leveraging Blockchain for secure law enforcement and government records”**. We have completed it as a part of the curriculum of Visvesvaraya Technological University, Belagavi in partial fulfillment of the requirements for the VII semester of Bachelor of Engineering in Computer Science and Engineering.

We are profoundly indebted to our guide, **Dr. Mustafa Basthikodi**, Professor & Head, Department of Computer Science and Engineering for innumerable acts of timely advice, encouragement and we sincerely express our gratitude.

We also thank **Dr. Suhas A Bhyratae** and **Ms. Prapulla G**, Project Coordinators, Department of Computer Science and Engineering for their constant encouragement and support extended throughout.

We express our sincere gratitude to **Dr. Mustafa Basthikodi**, Professor & Head, Department of Computer Science and Engineering for his invaluable support and guidance.

We sincerely thank **Dr. S. S. Injaganeri**, Principal, Sahyadri College of Engineering and Management, who have always been a great source of inspiration.

Finally, yet importantly, we express our heartfelt thanks to our family and friends for their wishes and encouragement throughout the work.

Himanshu S Shetty (4SF22CS078)
M Imaad Iqbal (4SF22CS111)
Pratheek G Shetty (4SF22CS148)
Shifali Florine Lobo (4SF22CS192)

TABLE OF CONTENTS

Abstract **i**

Acknowledgement **ii**

Table of Contents **v**

List of Figures **vii**

List of Code Snippets **viii**

1 Introduction **1**

1.1 Overview 1

1.2 Motivation 2

1.3 Scope of the Project 2

1.4 Definitions, Acronyms, and Abbreviations 3

1.5 Structure of the Report 4

2 Literature Survey **5**

2.0.1 Limitations 11

2.0.2 Research Gaps identified 12

2.1 Contributions of this Project Work 12

3 Problem Formulation **14**

3.1 Problem Statement 14

3.2 Problem Description 14

3.3 Objectives 15

3.4 Functional Requirements 15

3.5 Non-Functional Requirements 15

3.6 Software and Tools 16

4 Project Design and Implementation **17**

4.1 Architecture Diagram 17

4.2 Use Case Diagram 18
4.3 Data Flow Diagrams 19
4.4 Class Diagram 21
4.5 Sequence Diagram 23
4.6 Processing Pipelines 23
4.6.1 Evidence Upload and Secure Storage Pipeline 24
4.6.2 Evidence Retrieval and Verification Pipeline 26
4.6.3 Utility Methods 27
4.7 Core Services 32
4.7.1 Overview 32
4.7.2 Backend Service 33
4.7.3 Client Application 35
4.7.4 IPFS Service 36
4.7.5 Blockchain Network (Hyperledger Fabric) 37

5 Results and Discussion 41
5.1 Experimentation Details 41
5.1.1 Hardware Environment 41
5.1.2 Software and Network Architecture 42
5.2 Results Obtained 43
5.2.1 Functional Validation and test case validation 43
5.2.2 Performance Analysis for Scalability 45
5.3 Societal Impact of the Project 47
5.4 SDG (Sustainable Development Goals) Mapped 48
5.5 Discussions 50
5.5.1 Insights from the Study 50
5.5.2 Challenges and Problems Faced 51
5.6 Conclusion of results 51

6 Conclusion and Future Work 53
6.1 Conclusion 53
6.1.1 Summary of Findings 53
6.1.2 Significance and Contribution of the LEDGIS Framework 53
6.1.3 Final Remarks on Project Objectives 54
6.2 Future Enhancements 54
6.2.1 Architectural Enhancements: On-Chain Access Control 54
6.2.2 Performance and Scalability Improvements 55

6.2.3 Advanced Cryptography and Privacy Preservation 55
6.2.4 System Interoperability and Governance 56

References **64**

APPENDIX - A : Plagiarism Report Front Page (Project Report) **65**

APPENDIX - C : Paper Publication Details **66**

APPENDIX - C : Copy of Paper Published **68**

LIST OF FIGURES

4.1 Component-Based Architecture for Legal Blockchain Application 17

4.2 Use Case Diagram showing the interaction between users, backend server and the ledger. 18

4.3 Level 0 Data Flow Diagram: Overview of the evidence capture and storage system 19

4.4 Level 1 Data Flow Diagram: Breakdown of subsystems involved in evidence processing 19

4.5 Level 2 Data Flow Diagram: Detailed internal flow of hashing, encryption, and metadata storage 20

4.6 Class diagram illustrating the modular design of a blockchain-enabled access control system for cloud services. 21

4.7 Sequence diagram demonstrating the request-response workflow from the user to the blockchain network and cloud services. 23

4.8 Evidence Upload and Secure Storage Pipeline 24

4.9 Evidence Retrieval and Verification Pipeline 26

4.10 High-level overview of LEDGIS core services and their communication channels 33

4.11 Postman testing of the login route showing JWT generation upon successful authentication 34

4.12 Admin-only register route accessed via Postman using JWT in authorization header 34

4.13 LEDGIS client interface showing evidence upload and verification sections 35

4.14 IPFS daemon running locally showing peer discovery 36

4.15 Docker containers showing active Hyperledger Fabric peers, orderer, and CA services 37

5.1 Postman Test: Evidence Upload API returning success with hash and CID storage confirmation 44

5.2 Postman Test: Metadata Retrieval showing hash, key, IV, and chunk references from ledger 44
5.3 Global IPFS Map: A webpage showcasing the global ipfs storage system nodes along with the node statistics 45
5.4 Evidence Commit Page: Interface to submit new evidence to the ledger . 46
5.5 System processing time for handling files of various sizes, demonstrating a linear performance scaling that supports large files 46
5.6 A comparison of API response times, showing the higher latency of the ingestion process due to cryptographic and consensus overhead. 47
5.7 Sustainable Development Goals (SDGs) 50
5.8 System performance under concurrent load, illustrating the relationship between throughput and response time as user load increases 51

LIST OF CODE SNIPPETS

4.1 Chunking and reconstruction utilities (IPFS integration) 28
4.2 Encryption utilities (AES-256-CBC) 29
4.3 Hashing utilities (SHA-256) 30
4.4 EvidenceContract chaincode for storing and retrieving evidence metadata 38
4.5 Backend submission controller for interacting with Hyperledger Fabric . . 39

CHAPTER 1

INTRODUCTION

In the modern digital era, governmental and legal institutions increasingly rely on electronic data, yet traditional centralized record management systems remain inherently vulnerable to data breaches, manipulation, and single-point failures. These vulnerabilities pose significant risks to judicial integrity and public trust, prompting nations like Estonia and Georgia to explore blockchain as a more secure alternative. As cyber threats and bureaucratic inefficiencies rise, there is an urgent need to modernize legal infrastructures to guarantee the authenticity, security, and transparency of sensitive law enforcement records].

Blockchain technology, specifically the permissioned Hyperledger Fabric framework, offers a robust solution by providing a decentralized, tamper-proof ledger with fine-grained access control and automated smart contracts[cite: 106, 107]. This project investigates the application of such technology to design a secure architecture for managing government and legal records, ensuring data integrity while addressing practical challenges like scalability and compliance. By creating a system that protects against unauthorized alteration and supports controlled access, this work contributes to the evolution of digitally empowered governance and lays the groundwork for future innovations in legal technology.

1.1 Overview

The proposed system is built upon the Hyperledger Fabric blockchain framework, leveraging its modular architecture to support secure transaction processing, identity management, and smart contract execution. The system aims to store legal records in a tamper-proof ledger, where every access and modification request is logged and verifiable. While the blockchain stores metadata and transaction history, large documents and files are maintained in off-chain distributed storage to ensure scalability and efficient retrieval.

The system architecture is designed to support cooperation among multiple stakeholders such as courts, police departments, forensic labs, and legal representatives. Each participant in the network is assigned a unique identity with permissions defined through Membership Service Providers (MSP). This ensures that sensitive information is accessible only to authorized entities while preserving the transparency and integrity of the legal workflow. The system further integrates smart contracts to automate core processes like evidence submission, record validation, and authorization requests.

1.2 Motivation

The motivation behind this project stems from the increasing need for secure and trustworthy legal information systems. With growing instances of document forgery, cyberattacks, and manipulation of digital evidence, the credibility of legal proceedings is at risk. A system that ensures the authenticity and traceability of records is necessary to strengthen judicial integrity and institutional accountability.

Moreover, traditional paper-based and semi-digital record management systems often involve lengthy administrative procedures, delays, and human errors. Blockchain-based automation can reduce these inefficiencies significantly. By introducing a decentralized verification mechanism, the system promotes transparency, accountability, and trust among all stakeholders involved in the judicial process.

1.3 Scope of the Project

The scope of this project includes the design, implementation, and evaluation of a blockchain-based legal record management system using Hyperledger Fabric. The project focuses on:

- Developing a permissioned blockchain network with secure identity management.
- Creating smart contracts to automate legal document and evidence handling.
- Providing controlled access to authorized individuals and agencies.
- Maintaining immutability and traceability of document history.

- Integrating off-chain storage for scalability.

However, the project does not address large-scale national-level deployment, legal policy reform, or integration with legacy court management software beyond conceptual alignment. These areas are considered future extensions of the work.

1.4 Definitions, Acronyms, and Abbreviations

- **Hyperledger Fabric:** A permissioned blockchain framework designed for enterprise and government applications, offering modular architecture, privacy features, and scalable transaction processing.
- **MSP (Membership Service Provider):** A component in Hyperledger Fabric responsible for managing digital identities, authentication, and access control within the blockchain network.
- **Smart Contract / Chaincode:** Executable code deployed on the blockchain that defines and enforces business logic automatically when predefined conditions are met.
- **Ledger:** The shared database maintained collectively by all blockchain nodes, storing both the immutable transaction history and the current world state.
- **Off-chain Storage:** An external data storage system used to store large files while maintaining lightweight references or hashes on the blockchain to ensure integrity and traceability.
- **Chain-of-Custody:** A process that ensures the integrity, authenticity, and traceability of digital evidence from its creation to final use in legal or forensic contexts.
- **Hybrid On-chain/Off-chain Architecture:** A system design that stores critical metadata and integrity proofs on-chain, while large or sensitive data files are stored off-chain to balance performance and scalability.
- **IPFS (InterPlanetary File System):** A decentralized storage protocol used for off-chain storage and retrieval of files in a distributed network.
- **Peer Node:** A network participant in Hyperledger Fabric responsible for hosting ledgers, executing chaincode, and validating transactions.

- **Ordering Service:** A Hyperledger Fabric component that ensures transactions are properly ordered, batched, and distributed to peers for validation and commitment.
- **CouchDB:** A NoSQL database used as the state database in Hyperledger Fabric, storing the latest values of ledger data in a queryable JSON format.
- **TLS (Transport Layer Security):** A cryptographic protocol used to ensure secure communication between blockchain components and client applications.

1.5 Structure of the Report

The report is organized into multiple chapters. Chapter 1 provides an introduction to the problem context, motivation, scope, and conceptual foundation of the project. Chapter 2 discusses the literature review, highlighting existing systems and related work in the field of blockchain-based legal record management. Chapter 3 covers the system architecture, design considerations, and component-level descriptions. Chapter 4 explains implementation details including smart contract logic, network configuration, and integration mechanisms. Chapter 5 presents testing procedures, results, and evaluation metrics. Chapter 6 discusses conclusions, limitations, and directions for future research. Chapter 7 provides an overview of the project plan.

CHAPTER 2

LITERATURE SURVEY

In this section, we explore various existing blockchain solutions related to judicial and government record management. Our goal is to understand what has already been achieved, where these systems succeed, and importantly, where they fall short. This helps us identify the gaps that our work can address.

Liu and Zheng [1] developed a blockchain framework aimed at preserving judicial evidence with a focus on improving data integrity and traceability. Their approach strengthens the security and auditability of legal records. However, they didn't tackle the practical challenges involved in integrating such blockchain systems within actual government infrastructures, which is a significant hurdle.

Nakamoto [2] introduced the foundational concept of decentralized ledgers through Bitcoin, demonstrating trustless peer-to-peer transactions. While groundbreaking, its scope was purely financial and does not address the complexities of government or judicial record systems.

The Hyperledger Foundation [3] outlined multiple enterprise blockchain frameworks, including Fabric, which support modularity and privacy. While promising for public sector use, many of these remain underutilized in legal contexts outside pilot programs.

Zheng et al. [4] provided a technical overview of blockchain architectures, consensus protocols, and future trends. While their review is comprehensive, it lacks specific case studies on law enforcement or judicial deployments.

Tapscott and Tapscott [5] emphasized blockchain's potential to disrupt traditional systems, including governance. Though visionary, their insights remain largely strategic rather than technical or implementation-specific.

The Government of Estonia [6] implemented blockchain in its national e-governance systems, including e-residency and health data. Their success is well-documented but does not fully explore applications in judicial evidence or criminal records.

The Ministry of Justice in Georgia [7] tested blockchain in a land registry system. It improved transparency and reduced fraud but was limited to property records without exploring broader legal applications.

Risius and Spohrer [8] proposed a research framework for understanding blockchain adoption and its impact, identifying gaps in existing literature but offering few implementation strategies for public systems.

Atzori [9] explored decentralized governance models enabled by blockchain, suggesting potential for digital democracy. However, the study remained conceptual without addressing public sector constraints.

Kshetri [10] examined blockchain's role in enhancing cybersecurity and privacy. The findings are relevant to public records, but the study stops short of proposing judicial-specific use cases.

Yaga et al. [11] from NIST provided a government-oriented overview of blockchain, clarifying its potential and limitations. However, they offered few concrete case studies involving courts or legal enforcement.

Swan [12] presented blockchain as a foundation for a new digital economy. While insightful, the book remains general-purpose and lacks attention to public sector needs.

Sun et al. [13] proposed blockchain frameworks for smart cities, focusing on sharing economy models. Their ideas are adjacent but do not address legal or judiciary contexts.

Pilkington [14] introduced core blockchain principles and applications in digital transformation. Though applicable, there is limited emphasis on legal or forensic evidence use.

The European Commission [15] published a report on blockchain's potential for digital government. It suggests policy directions but lacks technical implementations.

Cong and He [16] discussed blockchain's potential to improve transparency and market fairness via smart contracts. While rigorous, their model is rooted in finance, not law or governance.

Zyskind et al. [17] built a privacy-preserving blockchain system for personal data control. Their emphasis on user sovereignty is relevant but diverges from institutional record-keeping.

Bhaskar and Chuen [18] explored the role of smart contracts in issuing verifiable digital certificates. This directly informs trust mechanisms in public documents, though the scale is limited.

Saito and Yamada [19] examined how blockchain redefines trust, offering a conceptual look at its transformative role. However, the paper does not transition from theory to real-world trials.

Yue et al. [20] demonstrated a blockchain framework for healthcare data sharing, emphasizing privacy and auditability. Though outside the judiciary, the architecture inspires ideas for forensic data sharing.

Kassen [21] proposed a blockchain-based model for lifelong, cross-referenced e-government services. It introduces a scalable architecture but lacks judiciary-specific modules.

Al Mamun et al. [22] reviewed blockchain-based EHR systems. Their emphasis on confidentiality and trust maps well onto the legal sector's needs for data protection.

Sousa [23] identified blockchain as a transformative force in the public sector. However, the analysis is focused on macro-level shifts, not technical deployments.

Khan et al. [24] analyzed Dubai's government blockchain initiatives. Their report showcases operational benefits and legal considerations, making it highly relevant for our context.

Tan et al. [25] proposed a framework for blockchain governance in the public sector. While conceptually strong, it lacks empirical validation through live deployments.

Kim et al. [26] created a privacy-preserving ledger framework for global human resource records. Its architecture suggests parallels with identity and role verification in legal systems.

Piao et al. [27] introduced a Service-on-Chain (SOC) model for secure government data sharing. Their approach offers a scalable and privacy-conscious design relevant for legal records.

Elisa et al. [28] designed a blockchain and artificial immunity-based e-government framework. It adds layers of AI-based trust assessment, though implementation remains in early stages.

Mahlaba et al. [29] proposed a secure document storage system to prevent corruption. Their model is tailored for sensitive documents, aligning closely with our project's goals.

Wang et al. [30] explored identity and data protection using blockchain trust models. This contributes to the broader conversation on secure authentication and access control.

Chen et al. [31] proposed a searchable-encryption enabled blockchain EHR sharing system that stores encrypted indexes on-chain and EHRs off-chain, enabling privacy-preserving search and owner-controlled access

Tang et al. [32] designed a cross-institution EMR sharing scheme combining searchable encryption with blockchain smart-contract ACLs to support patient-centric access and verifiable search.

Liu et al. [33] (BPDS) introduced a consortium-blockchain index plus cloud off-chain storage design using CP-ABE and content-extraction signatures for privacy-preserving EMR sharing.

Li & Han [34] (EduRSS) proposed anchoring hashes of off-chain encrypted educational records on-chain with smart contracts to automate sharing and integrity verification.

Naz et al. [35] implemented an Ethereum + IPFS prototype using RSA/SSS for encryption and smart contracts for auditable, incentive-driven data sharing.

Ullah et al. [36] presented an IoT-focused architecture using Ethereum + IPFS, AES for bulk encryption, ECDH for key exchange, and ABAC via smart contracts with PoA tuning for IoT constraints.

Sonkamble et al. [37] demonstrated a Hyperledger Fabric + IPFS solution with SPAKE key exchange and smart contracts for patient-centered EHR transmission, including empirical performance measurements.

Vidhya & Kalaivani [38] proposed a permissioned blockchain with smart-contract ACLs and an LFC encryption scheme for privacy-aware medical data sharing.

Verma & Kanrar [39] suggested combining attribute-based encryption (ABE) with blockchain metadata and smart contracts to enable fine-grained, off-chain document sharing.

Ma [40] proposed a covert document communication model combining Monero-inspired privacy techniques, IPFS, and ABE to enable stealthy encrypted transfers with traceable metadata.

Pandey et al. [41] presented an off-chain ABE framework with on-chain metadata and audit trails to improve throughput and traceability for securing digital documents.

Shyamala et al. [42] examined a private Ethereum (PoA) + AES-encrypted IPFS cluster approach with on-chain hashing to balance scalability and tamper-evidence for document storage.

Alruwaill et al. [43] (hChain) proposed edge+IoMT integration where edge devices pre-process and hash/encrypt EHRs, anchoring integrity on-chain and using smart contracts for sharing.

Siva Kumar et al. [44] introduced a sensitivity-aware encryption approach (RSFSA) that prioritizes protection of sensitive document fields within a blockchain-backed cloud storage model.

Zhang et al. [45] proposed a double-blockchain architecture to separate indexes from access logs and leverage ABE with off-chain encrypted EMRs for efficiency and security.

Kandpal [46] evaluated symmetric ciphers for serverless blockchain storage and recommended AES in permissioned serverless contexts for confidentiality and performance.

Zafar et al. [47] implemented a Hyperledger Fabric-based distributed framework for automotive supply-chain records, demonstrating feasibility and measuring memory/cost/update performance.

Kushch et al. [48] proposed a ”Blockchain Tree” multilevel design with subchains for hierarchical personal ID data storage and fine-grained access control.

Sai Sandeep & Yadlapalli [49] prototyped an Ethereum + IPFS document sharing system with smart-contract ACLs, frontend/backend integration, and planned layer-2 scalability considerations.

Recent work specifically addressing healthcare records and emergency access demonstrates concrete patterns that are directly relevant to government and judicial records management. Several systems propose storing encrypted medical records off-chain while anchoring searchable indexes or integrity hashes on-chain, combining searchable encryption and CP-ABE to enable privacy-preserving search and owner-controlled access [50, 51, 52, 53].

Prototypes and application papers explore Ethereum-based EHR implementations and smart-contract ACLs to automate sharing and auditing between institutions; these studies highlight practical integration and performance trade-offs for inter-hospital data exchange and emergency retrieval [54, 55, 56, 57].

Surveys and implementation reports also point to mobile and client-side considerations — for example, securing Android applications and enabling rapid emergency access while preserving confidentiality — and recommend combining robust symmetric encryption for bulk data with careful key-exchange and revocation mechanisms [58, 59].

Building on the works listed in our references, there is a substantial body of research addressing cloud storage and file-sharing systems that complement government and judicial record management. Several recent studies propose blockchain-backed frameworks to secure decentralized file sharing, integrating distributed storage protocols and access control primitives to reduce tamper risk and central points of failure [60, 61, 62].

Practical implementations explore combinations of blockchain with IPFS or cloud backends and evaluate encryption strategies and revocation mechanisms. For example, a system with attribute-based encryption integrated with smart contracts has been shown to provide fine-grained access and fast revocation in cloud file sharing scenarios [63], while serverless and cost-efficient designs examine tradeoffs between confidentiality, latency, and operational cost [62, 64, 65].

Specialized document verification systems such as Blockcerts illustrate applied architectures for issuing verifiable credentials and certificates at scale, offering practical patterns for government-grade certification and diploma verification [66]. These systems often combine on-chain anchors with off-chain storage and notarization APIs to balance performance and legal evidentiary needs [66, 67].

On the cryptographic and protocol side, survey papers and conference works review how blockchain can be combined with advanced privacy techniques — including proxy re-encryption, zero-knowledge proofs, and provenance mechanisms — to protect data provenance and cross-border identity management [68, 69, 70, 71]. These contributions identify concrete building blocks for privacy-preserving, auditable document sharing across institutional boundaries.

Several applied research pieces target specialized environments such as space-air-ground integrated networks (SAGIN) and IoT, demonstrating decentralized secure communication protocols tailored for high-latency or resource-constrained settings [72]. Broader surveys catalog privacy-focused blockchain applications and point to open problems in scalability, formal privacy guarantees, and interoperability [73, 74].

Collectively, these studies show that blockchain-anchored document security is feasible across domains (healthcare, cloud storage, legal, government), but recurring gaps remain: large-scale benchmarking, standardized metadata for portability across storage backends, robust key-revocation for ABE schemes, and a stronger regulatory mapping (GDPR/HIPAA) for production deployments [60, 63, 70].

2.0.1 Limitations

Based on the comprehensive survey, the limitations of existing work are manifold. Many proposed solutions remain highly theoretical or conceptual, focusing on the strategic potential of blockchain rather than on technical, implementation-specific details [5, 9]. Pioneering and well-documented government applications, such as those in Estonia [6] and Georgia [7], are often narrow in scope. They successfully validate blockchain for specific domains like e-residency or land registry but do not provide comprehensive models for the complex data flows, multi-actor permissions, and stringent evidentiary rules required by the judicial and law enforcement sectors. Furthermore, many academic reviews and frameworks lack specific case studies for law enforcement [4], fail to tackle the practical challenges of integration with legacy government infrastructure [1], or have not been empirically validated through live, integrated deployments [25].

2.0.2 Research Gaps identified

The identified limitations in the current literature point to a critical and persistent research gap: the need for a holistic, scalable, and empirically-validated framework engineered specifically for the legal and law enforcement domains. There is a distinct lack of solutions tailored for the unique forensic chain-of-custody rules and the complex, multi-actor access policies required by the judicial system (e.g., involving judges, prosecutors, defense attorneys, and forensic labs). A significant gap also exists in addressing the scalability challenge of storing large evidence files—such as high-definition video, audio recordings, or detailed forensic images—without compromising the integrity and performance of the blockchain ledger. Finally, while many systems demonstrate components like IPFS integration [35, 36] or Hyperledger Fabric use [37, 47], a complete, end-to-end system that integrates these components with domain-specific smart contracts and provides quantitative performance analysis is still needed.

2.1 Contributions of this Project Work

This project work titled "LEDGIS” addresses key challenges in the secure management of digital evidence and government records. It presents the design, implementation, and validation of a complete end-to-end framework using Hyperledger Fabric and IPFS to ensure both functional accuracy and scalability.

A major contribution of this work is the development and validation of a hybrid on-chain and off-chain architecture. In this model, lightweight and immutable metadata such as file hashes, AES encryption keys, initialization vectors, and IPFS content identifiers are stored on the Hyperledger Fabric ledger, while the large encrypted evidence files are stored off-chain in the IPFS network. This design improves scalability while maintaining verifiable data integrity.

Another contribution is the creation of domain-specific cryptographic pipelines designed for forensic use. Two complete pipelines were implemented and tested: the Evidence Upload Pipeline, which performs hashing, encryption, chunking, and storage on IPFS with metadata registration on the blockchain, and the Evidence Retrieval Pipeline, which reconstructs and decrypts data while verifying integrity through hash comparison. These pipelines ensure confidentiality and proof of integrity throughout the evidence lifecycle.

The project also introduces purpose-built smart contracts that automate legal and forensic workflows. Functions such as storeEvidence and getEvidence support the digital chain-of-custody and enforce evidence management logic within the blockchain network.

An integrated access control mechanism has been implemented using a role-based model at the backend API layer, which serves as a secure gateway to the permissioned blockchain. This ensures that only authorized users within the judicial and law enforcement ecosystem can perform specific operations, reflecting the real-world nature of forensic evidence management.

Finally, the project provides empirical performance validation through detailed testing and benchmarking. Quantitative evaluations of API latency, transaction throughput, and user load, as discussed in Chapter 6, demonstrate predictable performance and practical scalability. These results confirm the system's feasibility for real-world deployment within legal and government domains.

CHAPTER 3

PROBLEM FORMULATION

3.1 Problem Statement

Government legal records are critical assets that require secure, tamper-proof, and transparent handling. Traditional centralized record management systems are vulnerable to cyberattacks, unauthorized modifications, and data loss. These vulnerabilities undermine trust in judicial systems and delay legal proceedings. There is a pressing need for a secure, decentralized solution that ensures integrity, transparency, and traceability of legal documents.

3.2 Problem Description

The proposed project aims to address these issues by leveraging blockchain technology to build a secure and transparent framework for managing government legal records. Using Hyperledger Fabric, a permissioned blockchain platform, the system will ensure that access to legal data is strictly controlled and auditable. Smart contracts will automate document verification, timestamping, and access authorization, reducing human errors and procedural delays. The architecture includes a distributed ledger for tamper-proof recordkeeping, a client interface for user access, and secure APIs for system integration. The system also incorporates encryption services and identity management to safeguard sensitive data. This blockchain-based solution is expected to enhance efficiency, accountability, and public trust in legal and law enforcement institutions while addressing implementation challenges such as scalability, interoperability, and legal compliance.

3.3 Objectives

- Develop a Hyperledger Fabric-based system for managing law enforcement records securely.
- Create a pipeline to hash documents and store files off-chain with on-chain hash verification.
- Implement smart contracts to automate legal processes such as evidence validation and warrant issuance.
- Apply role-based access control and maintain audit logs for all actions within the system.
- Assess scalability, legal compliance, and integration with existing government systems.

3.4 Functional Requirements

- **Secure Record Creation:** The system must allow authorized government officials to create and upload legal records securely onto the blockchain.
- **Smart Contract Execution:** The system must support smart contracts for automating document approval, verification, and transfer of ownership.
- **Tamper-Proof Record Storage:** The system must ensure that once a record is written to the blockchain, it cannot be modified without consensus.
- **User Access Management:** The system must offer role-based access, allowing different levels of interaction for law enforcement, judicial staff, and public users.
- **Audit Trail Generation:** The system must generate immutable logs of all transactions for verification and legal traceability.

3.5 Non-Functional Requirements

- **Security and Privacy:** The system must implement end-to-end encryption and comply with data protection policies applicable to legal records.

- **Performance:** Blockchain transactions (e.g., record uploads and retrieval) must be processed within seconds to ensure practical usability.
- **Scalability:** The system must support increasing volumes of legal data, users, and transactions, and scale effectively across multiple government departments.
- **Availability:** The system should have high availability to ensure 24/7 access to legal data, particularly for law enforcement and judiciary needs.
- **Maintainability:** The codebase and infrastructure must be modular and well-documented to support updates, patching, and integration with legacy systems.
- **Interoperability:** The solution must support integration with existing digital case management systems, e-governance portals, and authentication services.

3.6 Software and Tools

- **Hyperledger Fabric (v2.5):** Used to build a permissioned blockchain network with support for modular components such as consensus and membership services.
- **JavaScript (Node.js v18+):** Serves as the primary programming language for developing backend logic, writing chaincode, and integrating with the Hyperledger SDK.
- **Docker:** Used to containerize network components including peers, orderers, and certificate authorities, ensuring consistency and ease of deployment.
- **fabric-sdk-node / fabric-ca-client / fabric-contract-api:** These Hyperledger Fabric libraries enable client-side interactions with the blockchain network, handle user enrollment and identity management, and define smart contract APIs.
- **ExpressJS / NextJS:** ExpressJS is used to build the RESTful backend APIs while NextJS powers the frontend interface with server-side rendering capabilities.
- **crypto-js:** Provides cryptographic functions such as hashing and AES encryption to ensure data integrity and confidentiality.
- **Encrypted File Storage (e.g., IPFS / Local Vaults):** Supports off-chain document storage for large or non-transactional data while preserving immutability and auditability.

CHAPTER 4

PROJECT DESIGN AND IMPLEMENTATION

4.1 Architecture Diagram

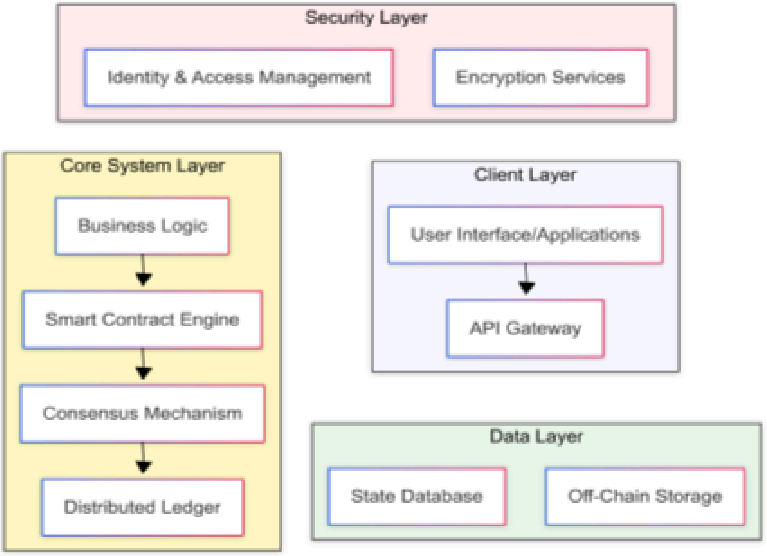


Figure 4.1: Component-Based Architecture for Legal Blockchain Application

The diagram illustrates the Component-Based Architecture for the legal blockchain application, structured into four primary layers: Security, Core System, Client, and Data. The Security Layer ensures data protection and controlled access through identity management and encryption services. It acts as the foundational safeguard for sensitive government and law enforcement data. The Core System Layer manages the essential blockchain operations, including business logic, smart contract execution, consensus mechanisms, and maintenance of the distributed ledger. Together, these elements establish the system's integrity and reliability.

The Client Layer provides interaction points for users and external systems through user interfaces and API gateways, ensuring smooth communication between the blockchain network and client applications. Finally, the Data Layer handles persistent data management with two key components: the on-chain State Database for structured records and Off-Chain Storage for large legal or forensic files. This modular design supports scalability, efficient resource management, and secure data handling, making it suitable for government and law enforcement record management systems.

4.2 Use Case Diagram

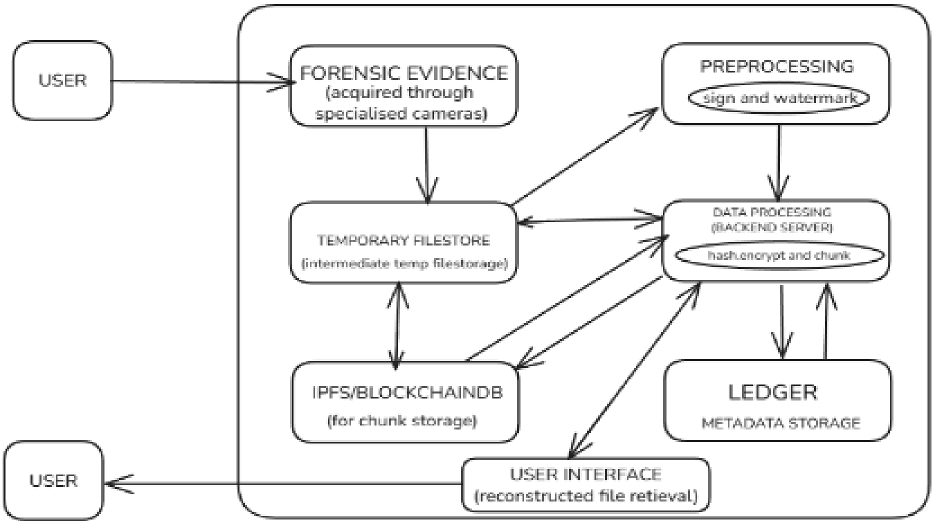


Figure 4.2: Use Case Diagram showing the interaction between users, backend server and the ledger.

This diagram illustrates the various actors and use cases in the system. End users can request access to legal records, view approved documents, and track the status of their requests. Administrators manage permissions and oversee approval workflows, while government agencies are responsible for uploading verified documents and auditing smart contract logs. The structure ensures transparency, accountability, and controlled access to sensitive data using blockchain technology.

4.3 Data Flow Diagrams

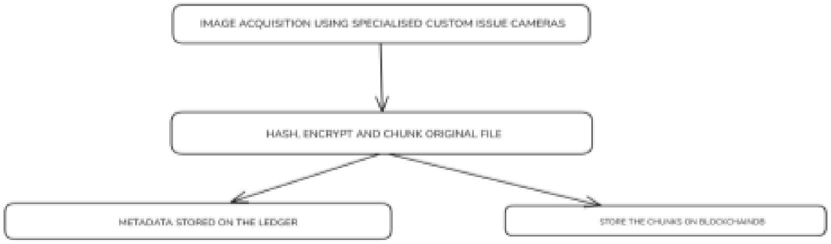


IMAGE ACQUISITION USING SPECIALISED CUSTOM ISSUE CAMERAS
HASH, ENCRYPT AND CHUNK ORIGINAL FILE
METADATA STORED ON THE LEDGER
STORE THE CHUNKS ON BLOCKCHAINDB

Figure 4.3: Level 0 Data Flow Diagram: Overview of the evidence capture and storage system

The Level 0 Data Flow Diagram provides a high-level overview of the system. It illustrates the basic interaction between users, the evidence capture mechanism, and the backend services responsible for storage and verification. The user submits evidence through a trusted interface, which is then securely processed and stored using blockchain and distributed storage technologies to ensure data integrity and immutability.

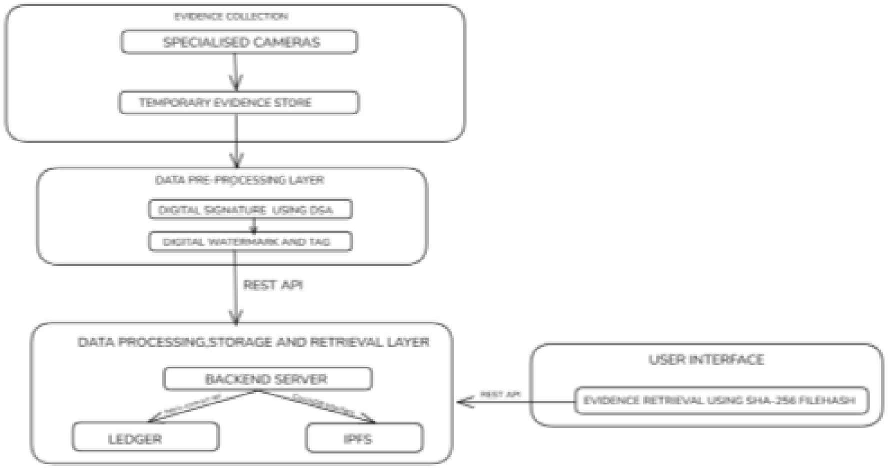

EVIDENCE COLLECTION
SPECIALISED CAMERAS
TEMPORARY EVIDENCE STORE
DATA PRE-PROCESSING LAYER
DIGITAL SIGNATURE USING DSA
DIGITAL WATERMARK AND TAG
REST API
DATA PROCESSING,STORAGE AND RETRIEVAL LAYER
BACKEND SERVER
LEDGER
IPFS
USER INTERFACE
REST API
EVIDENCE RETRIEVAL USING SHA-256 FILEHASH

Figure 4.4: Level 1 Data Flow Diagram: Breakdown of subsystems involved in evidence processing

The Level 1 Data Flow Diagram expands the high-level architecture into distinct subsystems: evidence acquisition, preprocessing, and secure storage. Evidence is collected using a specialised capture device and temporarily stored. It then undergoes digital signing and watermarking before being transferred to the backend. The backend handles communication with IPFS for storing encrypted file chunks and with the Hyperledger Fabric ledger for recording metadata.

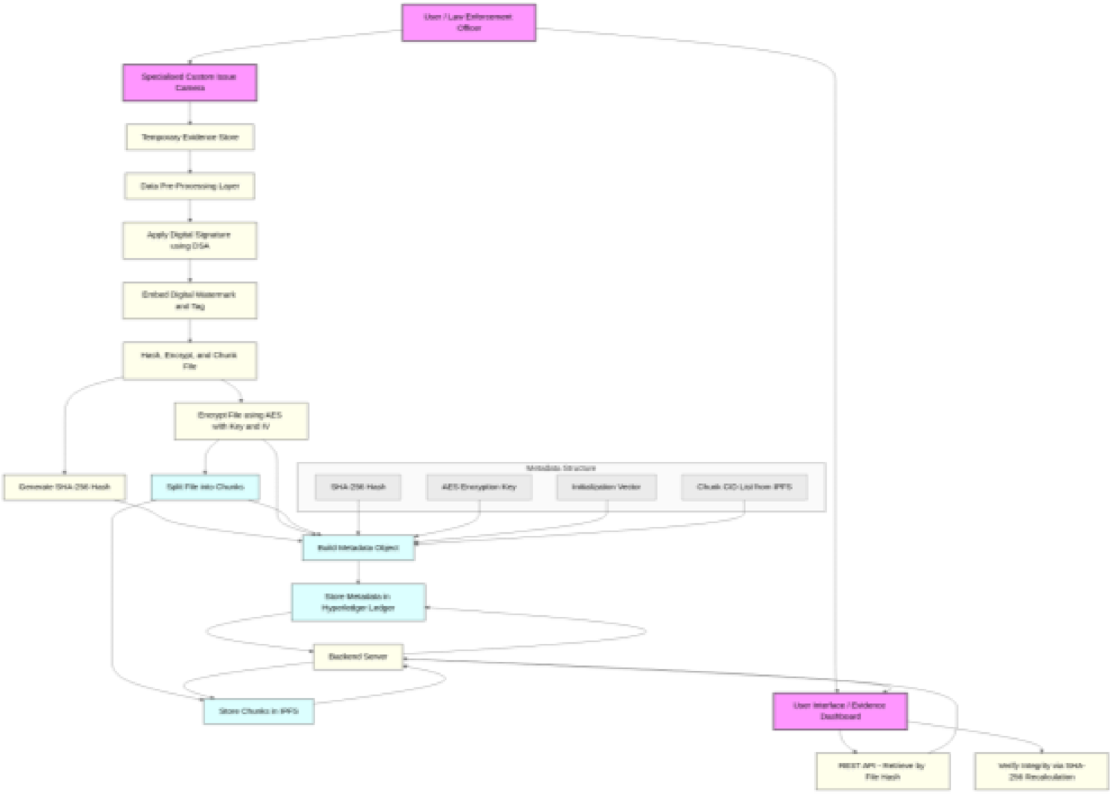

User / Law Enforcement Officer
Specialized Custom Issue Camera
Temporary Evidence Store
Data Pre-Processing Layer
Apply Digital Signature using DSA
Embed Digital Watermark and Tag
Hash, Encrypt, and Chunk File
Encrypt File using AES with Key and IV
Generate SHA-256 Hash
Split File into Chunks
Metadata Structure
SHA-256 Hash
AES Encryption Key
Initialization Vector
Chunk CID List from IPFS
Build Metadata Object
Store Metadata in Hyperledger Ledger
Backend Server
Store Chunks in IPFS
User Interface / Evidence Dashboard
REST API - Retrieve by File Hash
Verify Integrity via SHA-256 Recalculation

Figure 4.5: Level 2 Data Flow Diagram: Detailed internal flow of hashing, encryption, and metadata storage

The Level 2 Data Flow Diagram details the internal flow of evidence processing. After capture and preprocessing, the evidence is hashed using SHA-256, encrypted using AES with a unique key and IV, and then split into multiple chunks. These chunks are uploaded to IPFS, and their content identifiers (CIDs) are collected. A metadata object containing the file hash, AES key, IV, and chunk CIDs is created and stored on a Hyperledger ledger. This layered approach ensures traceability, tamper-evidence, and secure retrieval through REST-based queries and backend logic.

4.4 Class Diagram

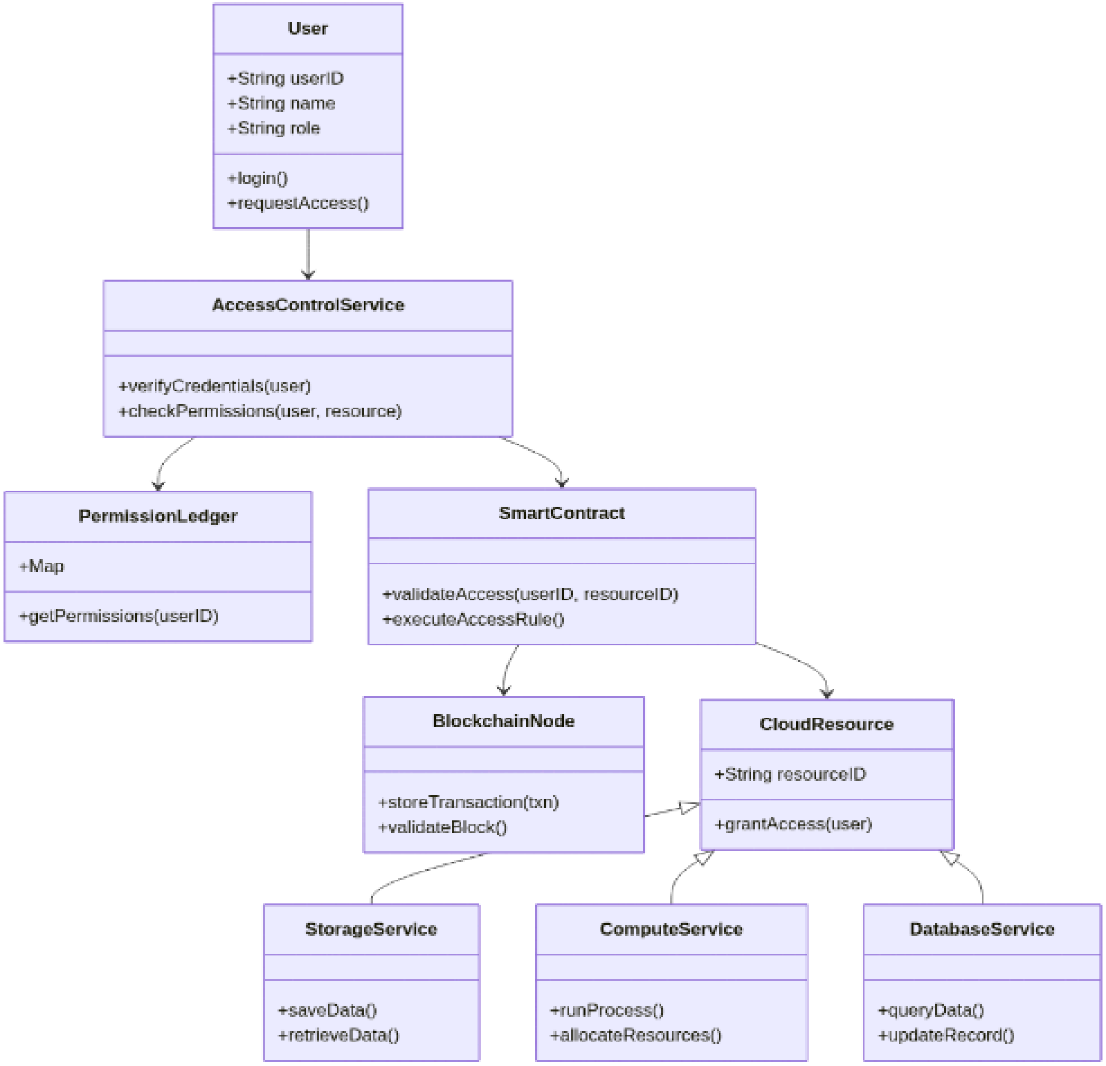


Figure 4.6: Class diagram illustrating the modular design of a blockchain-enabled access control system for cloud services.

This class diagram demonstrates the object-oriented architecture of a secure access control system built using blockchain technology. The system is composed of the following major components:

- **User:** Represents individuals attempting to access cloud resources. Attributes include `userID`, `name`, and `role`. Methods like `login()` and `requestAccess()` initiate interaction with the system.

- **AccessControlService:** Core service class responsible for verifying user credentials and checking resource permissions using the `verifyCredentials()` and `checkPermissions()` methods.
- **PermissionLedger:** Stores user permissions in a map structure and provides the method `getPermissions()` to retrieve access rights based on userID.
- **SmartContract:** Encapsulates blockchain logic for validating access (`validateAccess()`) and executing rules (`executeAccessRule()`). Operates autonomously on the blockchain.
- **BlockchainNode:** Responsible for recording validated access requests using `storeTransaction()` and ensuring data consistency through `validateBlock()`.
- **CloudResource:** Represents resources such as storage, compute, or database services. Grants access via `grantAccess()` and stores resource identifiers.
- **StorageService, ComputeService, DatabaseService:** Subclasses of CloudResource, each supporting domain-specific methods:
  - `StorageService: saveData()`, `retrieveData()`
  - `ComputeService: runProcess()`, `allocateResources()`
  - `DatabaseService: queryData()`, `updateRecord()`

The modular design enables clear separation of concerns, where user interactions, access validation, blockchain transaction management, and resource provisioning are handled independently. This architecture ensures scalability, auditability, and security in managing resource access in a distributed cloud environment.

4.5 Sequence Diagram

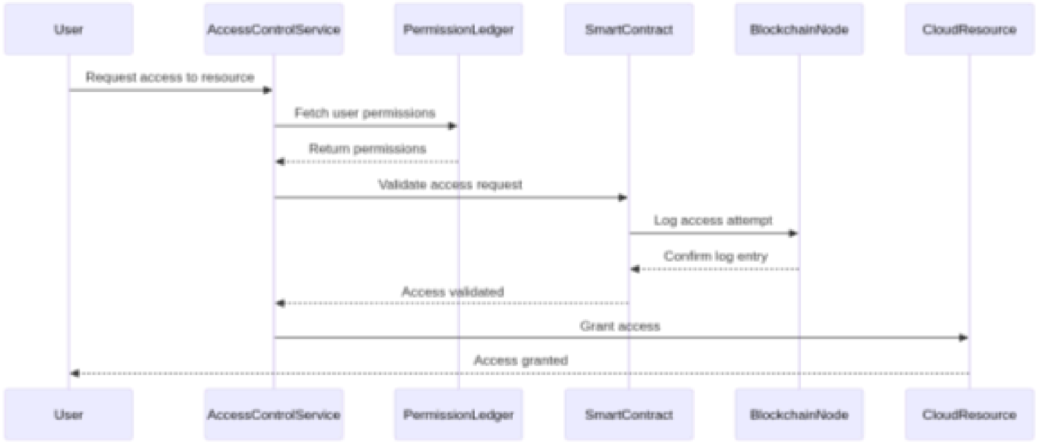


Figure 4.7: Sequence diagram demonstrating the request-response workflow from the user to the blockchain network and cloud services.

This sequence diagram illustrates the step-by-step flow of operations when a user attempts to access a protected resource. The request is first verified by the access control service, which then checks permissions and engages smart contracts on the blockchain for validation. Upon approval, the user is granted access to the requested cloud-based resource. This ensures all access activities are secure, traceable, and compliant with system policies.

4.6 Processing Pipelines

The LEDGIS system relies on two primary processing pipelines that work together to ensure every digital evidence file is stored, verified, and retrieved with complete security and transparency. These pipelines combine cryptography, distributed storage, and blockchain immutability to maintain both trust and accessibility in handling sensitive legal data.

4.6.1 Evidence Upload and Secure Storage Pipeline

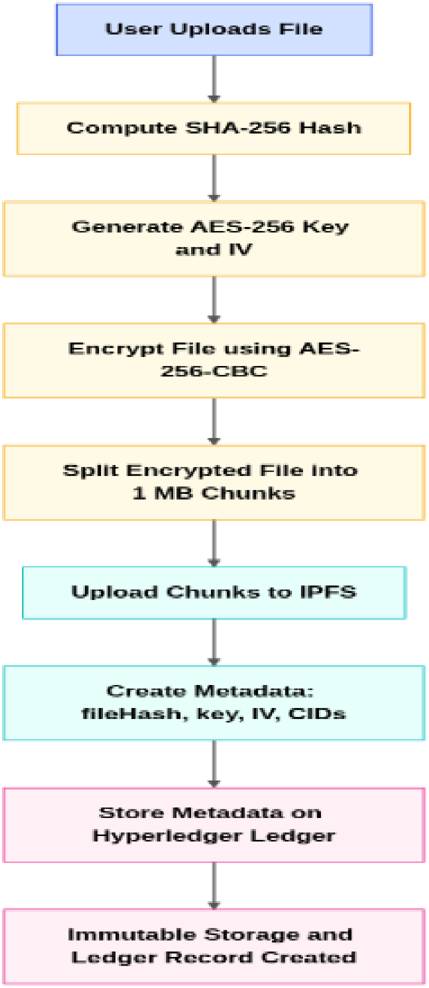

User Uploads File
Compute SHA-256 Hash
Generate AES-256 Key and IV
Encrypt File using AES-256-CBC
Split Encrypted File into 1 MB Chunks
Upload Chunks to IPFS
Create Metadata: fileHash, key, IV, CIDs
Store Metadata on Hyperledger Ledger
Immutable Storage and Ledger Record Created

Figure 4.8: Evidence Upload and Secure Storage Pipeline

When a user uploads a new piece of evidence, the system immediately initiates a secure end-to-end process to protect and register it. The workflow proceeds as follows:

1. **Hashing the File:** The uploaded file is hashed using the SHA-256 algorithm. This hash acts as a unique digital fingerprint — even the slightest change in the file would produce a completely different hash, making tampering easily detectable.

2. **Generating Encryption Keys:** A random AES-256 encryption key and a 128-bit initialization vector (IV) are generated for each upload. These values are never reused, ensuring strong protection for every individual file.

3. **Encrypting the Evidence:** The file is then encrypted using AES-256 in CBC mode. This step ensures that even if someone gains access to the storage layer, they cannot view or interpret the content without the correct key and IV.

4. **Chunking and IPFS Upload:** The encrypted file is split into 1 MB chunks and uploaded to the IPFS (InterPlanetary File System) network. IPFS returns a set of unique content identifiers (CIDs), which serve as permanent references to each chunk across the distributed network.

5. **Recording Metadata on Blockchain:** Finally, a metadata object containing the file's hash, AES key, IV, and list of CIDs is created. This metadata is stored on the Hyperledger Fabric ledger, ensuring an immutable and verifiable record of the evidence.

This pipeline guarantees that every uploaded file is encrypted, traceable, and permanently verifiable without ever compromising user privacy. The blockchain serves as the trust anchor, while IPFS provides efficient, decentralized storage.

4.6.2 Evidence Retrieval and Verification Pipeline

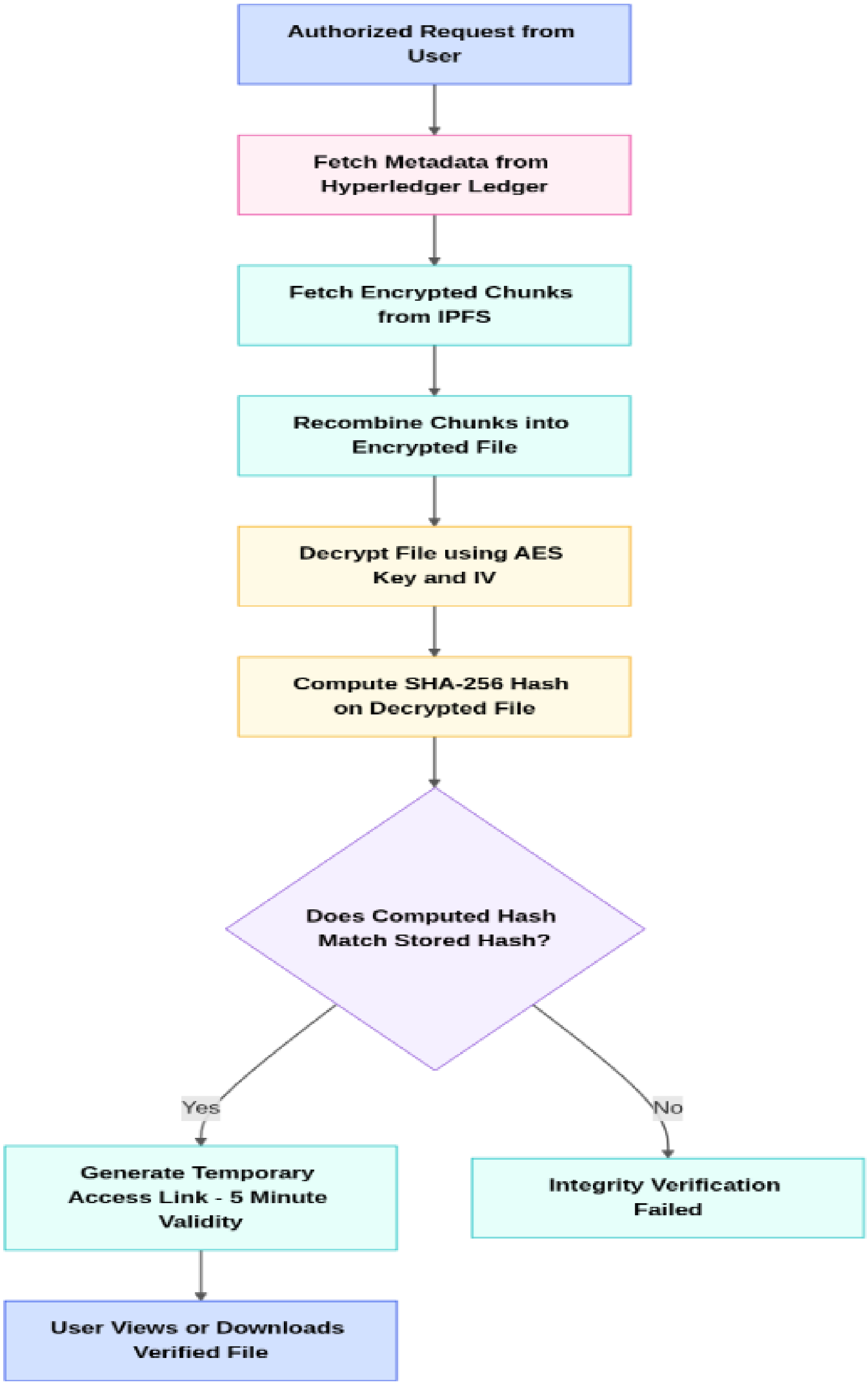

Authorized Request from User
Fetch Metadata from Hyperledger Ledger
Fetch Encrypted Chunks from IPFS
Recombine Chunks into Encrypted File
Decrypt File using AES Key and IV
Compute SHA-256 Hash on Decrypted File
Does Computed Hash Match Stored Hash?
Yes
No
Generate Temporary Access Link - 5 Minute Validity
Integrity Verification Failed
User Views or Downloads Verified File

Figure 4.9: Evidence Retrieval and Verification Pipeline

When an authorized user requests access to a stored file, the system activates a second pipeline that focuses on secure retrieval and authenticity verification. The process unfolds as follows:

1. **Fetching Metadata:** The system queries the Hyperledger Fabric ledger to retrieve the stored metadata, which includes the file's original hash, encryption key, IV, and list of IPFS chunk identifiers.

2. **Reconstructing the File:** Using the CIDs, the system locates and fetches the encrypted chunks from the IPFS network. The chunks are combined in sequence to reconstruct the original encrypted file.

3. **Decrypting the File:** The AES key and IV from the metadata are used to decrypt the reconstructed file, restoring it to its original form exactly as it was uploaded.

4. **Verifying Integrity:** A new SHA-256 hash is computed on the decrypted file and compared against the original hash from the ledger. If they match, the system confirms that the file has not been altered in any way since it was first stored.

5. **Providing Temporary Access:** Once verified, the system generates a secure, time-limited download link that remains valid for only five minutes. This ensures evidence can be accessed conveniently, but never stays exposed beyond necessity.

This retrieval pipeline not only confirms that a file is authentic and untampered but also enforces strong access control and privacy. It enables authorized parties to verify digital evidence with complete confidence without ever needing to trust a single centralized entity.

Together, these two pipelines form the operational core of LEDGIS. They ensure that every interaction with digital evidence from submission to retrieval remains secure, verifiable, and fully auditable, reflecting the project's goal of bringing trust and transparency to digital legal processes.

4.6.3 Utility Methods

The following utility modules provide the core file-processing primitives used by the LEDGIS pipelines: chunking and reconstruction (IPFS), encryption/decryption (AES-256-CBC), and hashing/verification (SHA-256). Each listing below is included verbatim from the implementation and is followed by a short explanation and usage notes.

```
const fs = require(’fs’);
const path = require(’path’);
const fse = require(’fs-extra’);
const {create}=require("ipfs-http-client")
const ipfs=create({ url: process.env.ipfsURL });
async function chunkFile(fileHash,filePath, chunkSizeMB, outputDir) {
    const chunkSize = chunkSizeMB * 1024 * 1024;
    const fileStream = fs.createReadStream(filePath, { highWaterMark: chunkSize });
    const chunkCIDs=[]
    let part=0;
    for await(const chunk of fileStream) {
        const result=await ipfs.add(chunk,{pin:true,rawLeaves:true});
        chunkCIDs.push({index: part,cid:result.cid.toString(),size: chunk.length});
        part++;
    }
    console.log(‘${part} chunks added to IPFS‘)
    console.log(chunkCIDs)
    return chunkCIDs;
}
async function reconFile(chunkCIDs, outputFilePath) {
    const writeStream = fs.createWriteStream(outputFilePath);
    chunkCIDs.sort((a,b)=>a.index-b.index)
    for(const chunks of chunkCIDs){
        const {cid}=chunks
        for await(const chunk of ipfs.cat(cid)){
            writeStream.write(chunk);
        }
    }
    writeStream.end();
    return new Promise((resolve)=>{
    writeStream.on(’finish’,()=>{
        console.log(‘Chunks combined into: ${outputFilePath}‘);
        resolve(outputFilePath);
    });
})}
module.exports={chunkFile,reconFile,ipfs};
```

Listing 4.1: Chunking and reconstruction utilities (IPFS integration)

Explanation — Chunking and Reconstruction

- `ipfs`: an `ipfs-http-client` instance created from `process.env.ipfsURL`. The environment variable must point to a reachable IPFS API endpoint (for example, `http://127.0.0.1:5001` in dev).
- `chunkFile(fileHash, filePath, chunkSizeMB, outputDir)`: reads the target file as a stream and splits it into chunks of size `chunkSizeMB` megabytes. For each chunk it calls `ipfs.add(..., {pin:true, rawLeaves:true})` and collects an array of objects `{index, cid, size}`. The function returns this `chunkCIDs` array which is meant to be stored in metadata (for example, on-chain).
- `reconFile(chunkCIDs, outputFilePath)`: takes the saved `chunkCIDs` (an array with indices and CIDs), sorts it by index, iterates over each CID and streams the chunk data back from IPFS using `ipfs.cat(cid)`. It writes each chunk in order to a file at `outputFilePath` and resolves with that path once the write stream finishes.
- **Usage notes:** Ensure `process.env.ipfsURL` is set and the IPFS daemon or gateway allows API calls. The functions stream data to avoid loading entire files into memory, so they are suitable for large evidence files. The returned `chunkCIDs` array is the canonical mapping you should include in the blockchain metadata so files can be reconstructed deterministically.

```
const crypto = require(’crypto’);
const fs = require(’fs’);
const algorithm = ’aes-256-cbc’;
const key = crypto.randomBytes(32);
const iv = crypto.randomBytes(16);
const { pipeline } = require(’stream/promises’);
function encryptFile(inputPath, outputPath) {
    const cipher = crypto.createCipheriv(algorithm, key, iv);
    const input = fs.createReadStream(inputPath);
    const output = fs.createWriteStream(outputPath);

    return new Promise((resolve, reject) => {
        input.pipe(cipher).pipe(output)
            .on(’finish’, () => resolve({ key, iv }))
            .on(’error’, reject);
    });
}
async function decryptFile(inputPath, outputPath, keyHex, ivHex) {
    const key = Buffer.from(keyHex, 'hex');
    const iv = Buffer.from(ivHex, 'hex');
    const decipher = crypto.createDecipheriv('aes-256-cbc', key, iv);
    try{await pipeline(fs.createReadStream(inputPath),decipher,fs.
       createWriteStream(outputPath));}
    catch(err){
        console.log(err)
    }
}


module.exports = { encryptFile,decryptFile, key, iv };
```

Listing 4.2: Encryption utilities (AES-256-CBC)

Explanation — Encryption / Decryption

- `algorithm, key, iv`: the module uses AES-256-CBC. `key` and `iv` are generated at module load using `crypto.randomBytes`.
- `encryptFile(inputPath, outputPath)`: creates a cipher stream and pipes the input file through it into the output file. The returned promise resolves with an object {`key, iv`} (Buffers) once the encrypted file is written.
- `decryptFile(inputPath, outputPath, keyHex, ivHex)`: accepts hex-encoded key and IV strings, converts them to Buffers, and uses a decipher stream to restore the plaintext file into `outputPath`. It uses `stream/promises.pipeline` to handle backpressure and streaming errors.

```
const crypto = require('crypto');
const fs = require('fs');

function hashFile(filePath) {
    return new Promise((resolve, reject) => {
        const hash = crypto.createHash('sha256');
        const stream = fs.createReadStream(filePath);
        stream.on('data', data => hash.update(data));
        stream.on('end', () => resolve(hash.digest('hex')));
        stream.on('error', reject);
    });
}

async function verifyHash(filePath, expectedHash) {
    const hash = crypto.createHash('sha256');
    const fileBuffer = fs.readFileSync(filePath);
    hash.update(fileBuffer);

    const calculatedHash = hash.digest('hex');
    if (calculatedHash === expectedHash) return true;
    return false;
}


module.exports = {hashFile,verifyHash};
```

Listing 4.3: Hashing utilities (SHA-256)

Explanation — Hashing and Verification

- `hashFile(filePath)`: streams the file and computes its SHA-256 digest, returning the hex string. Streaming keeps memory usage low for large files.

- `verifyHash(filePath, expectedHash)`: reads the whole file into memory and computes its SHA-256 value synchronously, then compares it to the expected hash and returns a boolean. For very large files you may prefer a streaming comparison (like `hashFile`) to avoid loading the entire file into memory.

- **Usage notes:** Use `hashFile` to create the canonical fingerprint before encrypting (or after decrypting, as part of verification). Store the produced hex hash in the ledger metadata. During retrieval, after decrypting the reassembled file, call `verifyHash` (or re-run `hashFile` and compare) to confirm integrity.

Integration notes and best practices

- Typical upload flow: `hashFile` → `encryptFile` (persist key/iv hex) → `chunkFile` (store returned `chunkCIDs`) → store metadata on-chain (hash, key hex, iv hex, chunkCIDs).

- Typical retrieval flow: fetch metadata from ledger → `reconFile` (download chunks, reconstruct encrypted file) → `decryptFile` (with stored key/iv) → `verifyHash`.

- Protect keys: keep AES keys and IVs confidential and consider wrapping them with an additional public-key envelope or a secrets manager for production deployments.
- IPFS availability and pinning: the code pins chunks on add (`pin:true`), but ensure long-term availability by running your own IPFS pinning service or using a pinning provider.
- Error handling: the listings are verbatim; consider adding higher-level retry/backoff and clearer error propagation in the calling code (API layer) to handle transient IPFS or I/O failures gracefully.

Together, these utility methods implement the low-level IO, cryptography, and storage primitives used by the LEDGIS processing pipelines. They are written to stream large files efficiently and to produce the canonical metadata (hashes, key/iv, and chunk CIDs) required for secure, auditable evidence management.

4.7 Core Services

4.7.1 Overview

The LEDGIS system is built as a distributed application consisting of multiple independent services each focused on a single responsibility. Instead of relying on a monolithic backend, LEDGIS separates concerns across a coordinated set of components that together provide a secure, auditable, and scalable solution for managing digital evidence.

Each service operates independently but communicates through well-defined APIs. The system comprises:

- A **backend server** that handles authentication, routing, cryptographic operations, and coordination between the blockchain and IPFS.
- A **client interface** that serves as GUI to interact with LEDGIS.
- The **IPFS network** which stores encrypted file chunks across distributed nodes.
- The **Hyperledger Fabric network**, which acts as the blockchain backbone for the metadata.

This modular architecture ensures that each service can be scaled, monitored, and updated independently, while still functioning cohesively within the system. Figure 4.10 provides a high-level view of how these services communicate with one another.

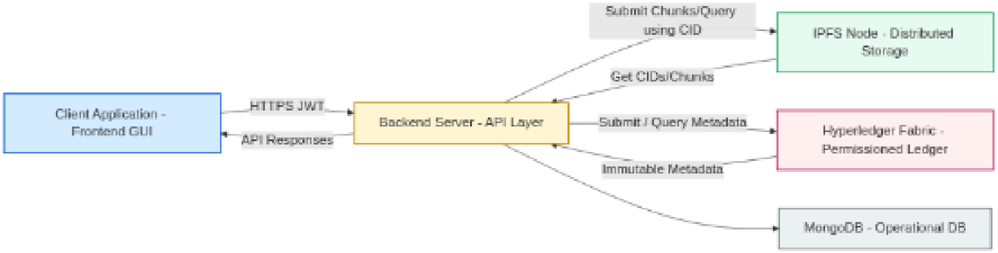

Client Application - Frontend GUI
HTTPS JWT
API Responses
Backend Server - API Layer
Submit Chunks/Query using CID
Get CIDs/Chunks
IPFS Node - Distributed Storage
Submit / Query Metadata
Immutable Metadata
Hyperledger Fabric - Permissioned Ledger
MongoDB - Operational DB

Figure 4.10: High-level overview of LEDGIS core services and their communication channels

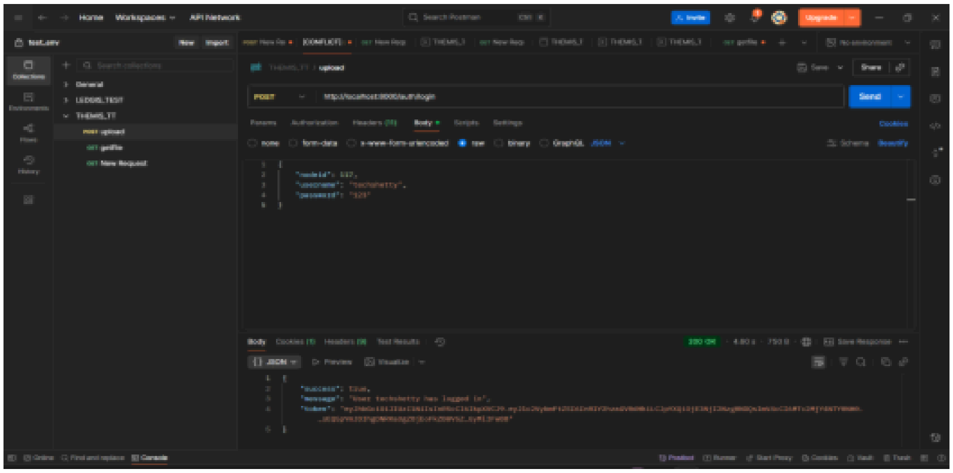
4.7.2 Backend Service

The backend server is the central coordination point for all major operations in LEDGIS. It is responsible for authentication, user management, evidence processing, communication with the blockchain, and interactions with IPFS. All critical workflows — such as the evidence upload and verification pipelines — are implemented as helper methods within the backend.

Authentication and Role-Based Access Control: The backend uses **JWT (JSON Web Tokens)** for secure, stateless authentication. Every route in the application is protected using custom middleware functions that validate the JWT token provided in the ‘Authorization‘ header. This ensures that only authenticated and authorized users can access system endpoints.

The system supports two user roles, ‘Admin‘ and ‘Regular‘. The ‘register‘ route, for example, is restricted exclusively to ‘Admin‘ users. Only an admin can register a new user, and this is enforced by verifying the JWT role claims in the middleware layer.

During development, all routes were tested through **Postman**, where the authentication process could be observed directly. Figure 4.11 and Figure 4.12 show successful responses from login and register routes.



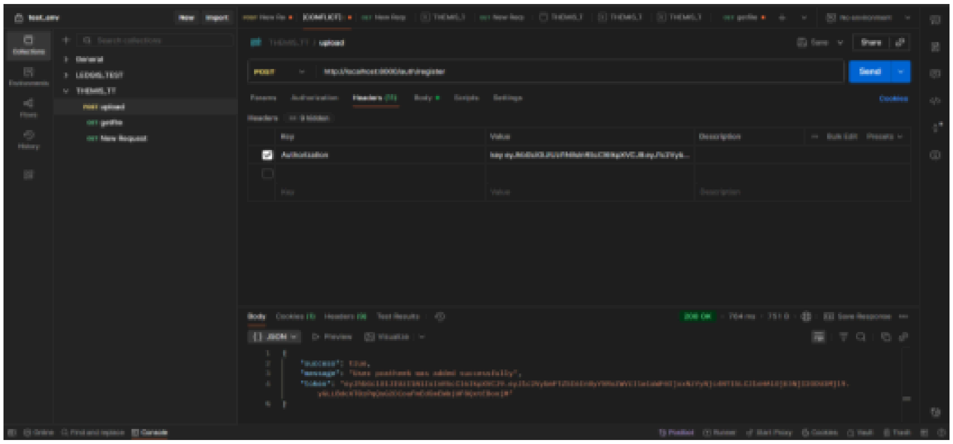
POST http://localhost:8000/auth/login

```
{
    "nodeId": 127,
    "username": "techshetty",
    "password": "123"
}
```

200 OK

```
{
    "success": true,
    "message": "User techshetty has logged in",
    "token": "[illegible]"
}
```

Figure 4.11: Postman testing of the login route showing JWT generation upon successful authentication



POST http://localhost:8000/auth/register

Key	Value	Description
☑ Authorization	key ey[illegible]…	

200 OK

```
{
    "success": true,
    "message": "User postbank was added successfully",
    "token": "[illegible]"
}
```

Figure 4.12: Admin-only register route accessed via Postman using JWT in authorization header

Protected Routes and Middlewares: Each backend endpoint is shielded using role-based middlewares that verify a user's identity and privileges before allowing further execution. Unauthorized users attempting to access sensitive endpoints receive descriptive error responses, maintaining transparency and clarity in API behavior.

Pipeline Execution: The backend also defines the two major pipelines as modular helper functions:

- **Evidence Upload Pipeline:** handles hashing, encryption, chunking, IPFS uploads, and blockchain metadata creation.

- **Evidence Retrieval Pipeline:** fetches metadata from the ledger, retrieves chunks from IPFS, reassembles the file, decrypts it, and verifies its hash.

These pipelines can be invoked through their respective API routes, ensuring complete automation of file integrity management within the backend.

Blockchain and IPFS Communication: The backend interacts with the **Hyperledger Fabric network** via the Fabric Contract API, allowing it to submit and query transactions programmatically. For file storage, it uses the IPFS HTTP client to upload and retrieve encrypted file chunks from the distributed network.

4.7.3 Client Application

The client application provides the graphical interface for all user interactions. Built with modern web technologies, it serves as the bridge between users and the backend API. The client does not perform any heavy computation or cryptography; instead, it focuses entirely on providing a clean and responsive interface.

A key security feature of the client is its `/check_reg` route verification mechanism. Whenever a user attempts to access a protected page (for instance, the evidence upload or dashboard view), the client automatically triggers a background API call to this route. The backend verifies the user's JWT and returns the access status. If the token is invalid or expired, the client redirects the user to the login page.

This small design detail prevents unauthorized users from accessing restricted components of the web application even through URL manipulation.

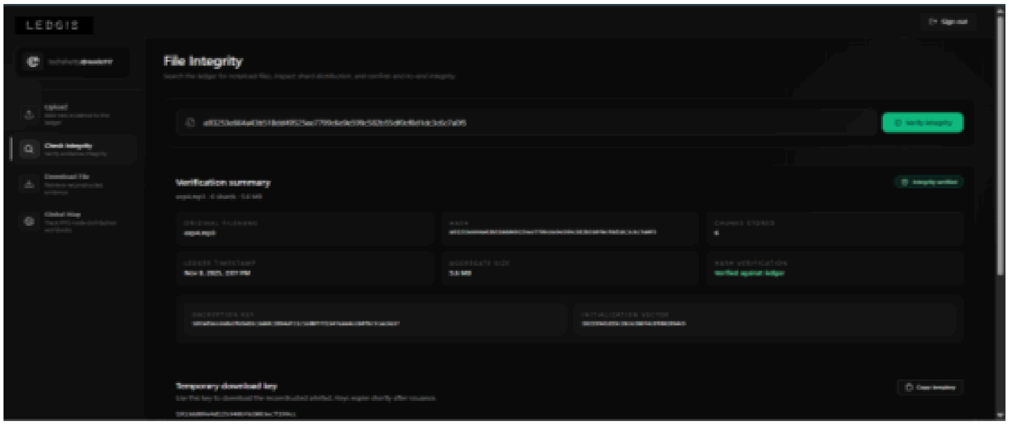


Figure 4.13: LEDGIS client interface showing evidence upload and verification sections

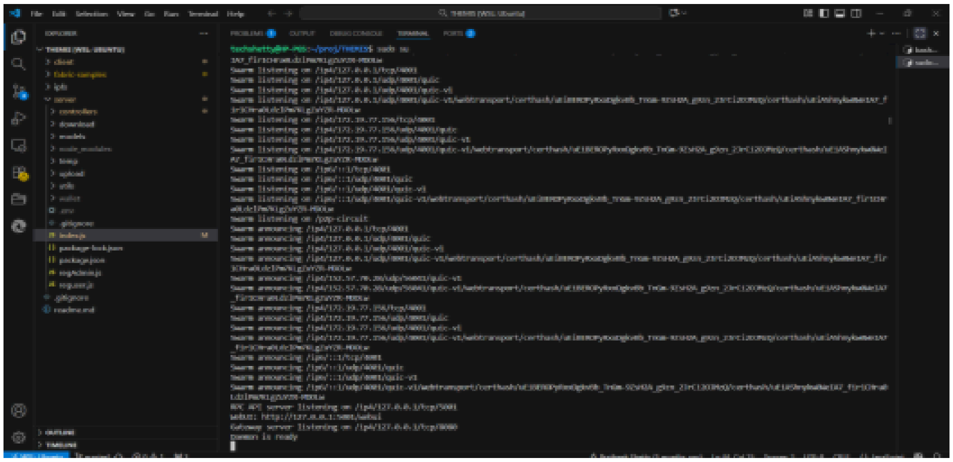
The client thus acts as a secure window into the distributed ecosystem, ensuring every action is validated and every data flow originates from an authenticated source.

4.7.4 IPFS Service

The InterPlanetary File System (IPFS) acts as LEDGIS's distributed storage layer. It eliminates the need for centralized cloud storage and ensures that encrypted evidence files remain immutable and accessible through content addressing. When the backend

uploads encrypted chunks, the IPFS daemon running on the server processes them and returns unique **Content Identifiers (CIDs)**. Each CID represents the SHA-256 hash of that chunk, meaning even a single byte change would alter its identifier completely. The backend maintains a connection to the IPFS API (available on port 5001), sending

upload and retrieval requests through HTTP endpoints. During operation, the daemon logs events such as peer connections, file pinning, and CID generation.



```
[illegible]:~/proj/THEMIS$ sudo su
[illegible]
Swarm listening on /ip4/127.0.0.1/tcp/4001
Swarm listening on /ip4/127.0.0.1/udp/4001/quic
Swarm listening on /ip4/127.0.0.1/udp/4001/quic-v1
Swarm listening on /ip4/127.0.0.1/udp/4001/quic-v1/webtransport/certhash/[illegible]
Swarm listening on /ip4/172.19.77.156/tcp/4001
Swarm listening on /ip4/172.19.77.156/udp/4001/quic
Swarm listening on /ip4/172.19.77.156/udp/4001/quic-v1
Swarm listening on /ip4/172.19.77.156/udp/4001/quic-v1/webtransport/certhash/[illegible]
Swarm listening on /ip6/::1/tcp/4001
Swarm listening on /ip6/::1/udp/4001/quic
Swarm listening on /ip6/::1/udp/4001/quic-v1
Swarm listening on /ip6/::1/udp/4001/quic-v1/webtransport/certhash/[illegible]
Swarm listening on /p2p-circuit
Swarm announcing /ip4/127.0.0.1/tcp/4001
Swarm announcing /ip4/127.0.0.1/udp/4001/quic
Swarm announcing /ip4/127.0.0.1/udp/4001/quic-v1
Swarm announcing /ip4/127.0.0.1/udp/4001/quic-v1/webtransport/certhash/[illegible]
Swarm announcing /ip4/152.57.78.20/udp/[illegible]/quic-v1
Swarm announcing /ip4/152.57.78.20/udp/[illegible]/quic-v1/webtransport/certhash/[illegible]
Swarm announcing /ip4/172.19.77.156/tcp/4001
Swarm announcing /ip4/172.19.77.156/udp/4001/quic
Swarm announcing /ip4/172.19.77.156/udp/4001/quic-v1
Swarm announcing /ip4/172.19.77.156/udp/4001/quic-v1/webtransport/certhash/[illegible]
Swarm announcing /ip6/::1/tcp/4001
Swarm announcing /ip6/::1/udp/4001/quic
Swarm announcing /ip6/::1/udp/4001/quic-v1
Swarm announcing /ip6/::1/udp/4001/quic-v1/webtransport/certhash/[illegible]
RPC API server listening on /ip4/127.0.0.1/tcp/5001
WebUI: http://127.0.0.1:5001/webui
Gateway server listening on /ip4/127.0.0.1/tcp/8080
Daemon is ready
```

Figure 4.14: IPFS daemon running locally showing peer discovery

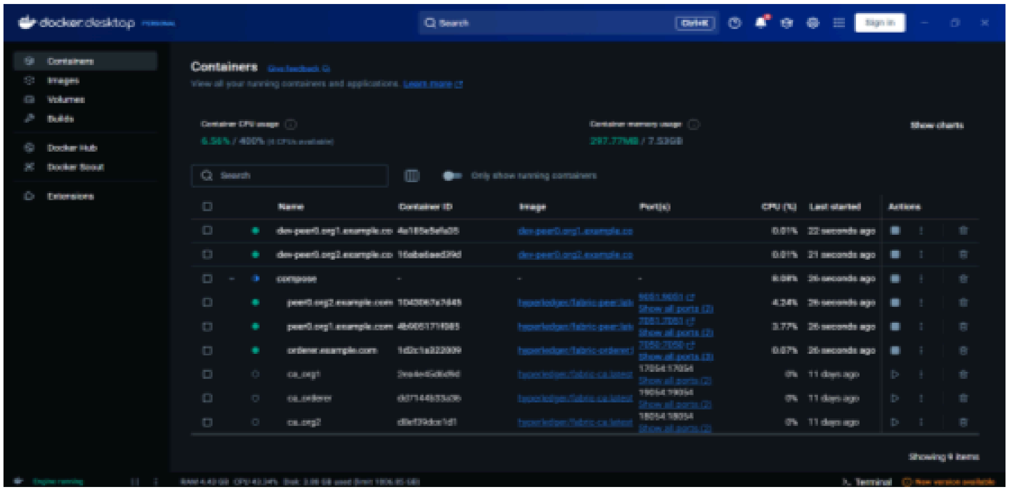
This screenshot (Figure 4.14) captures the IPFS daemon during active evidence uploads. The daemon continuously connects to other peers, forming a decentralized web of storage nodes. This ensures that evidence remains recoverable even if one node becomes unavailable due to its distributed nature.

4.7.5 Blockchain Network (Hyperledger Fabric)

The blockchain service, built on **Hyperledger Fabric**, is the backbone of LEDGIS's integrity and auditability. It functions as the permanent ledger for storing all evidence metadata — including the original file hash, encryption key, IV, and list of CIDs.

Unlike public blockchains, Fabric operates as a permissioned network. Only authenticated peers belonging to registered organizations can participate. This makes it ideal for a legal environment where privacy, traceability, and controlled access are critical.

All communications with the blockchain network occur through the backend server using the **Fabric Contract API**. The backend acts as a gateway between the application layer and the blockchain layer, handling all transaction submissions, queries, and event notifications. When a new piece of evidence is uploaded, the backend constructs a transaction proposal that includes the file hash, encryption metadata, and list of IPFS CIDs. This proposal is then endorsed by peers from all participating organizations before being ordered and committed to the ledger as a new block.



Containers

View all your running containers and applications. Learn more

Container CPU usage: 6.56% / 400% (4 CPUs available)

Container memory usage: 297.77MB / 7.53GB

	Name	Container ID	Image	Port(s)	CPU (%)	Last started
●	dev-peer0.org1.example.co	4a185e5efa35	dev-peer0.org1.example.co		0.01%	22 seconds ago
●	dev-peer0.org2.example.co	16abebaed39d	dev-peer0.org2.example.co		0.01%	21 seconds ago
●	compose	-	-	-	8.08%	26 seconds ago
●	peer0.org2.example.com	10d2067a7d45	hyperledger/fabric-peer:lat	9051:9051 Show all ports (2)	4.24%	26 seconds ago
●	peer0.org1.example.com	4b9051719385	hyperledger/fabric-peer:lat	7051:7051 Show all ports (2)	3.77%	26 seconds ago
●	orderer.example.com	1d2c1a222009	hyperledger/fabric-orderer:l	7050:7050 Show all ports (3)	0.07%	26 seconds ago
○	ca_org1	3ea4e45d6d9d	hyperledger/fabric-ca:latest	17054:17054 Show all ports (2)	0%	11 days ago
○	ca_orderer	dd7144b33a36	hyperledger/fabric-ca:latest	19054:19054 Show all ports (2)	0%	11 days ago
○	ca_org2	d8ef39dce1d1	hyperledger/fabric-ca:latest	18054:18054 Show all ports (2)	0%	11 days ago

Showing 9 items

Engine running RAM 4.42 GB CPU 43.34% Disk [illegible]

Figure 4.15: Docker containers showing active Hyperledger Fabric peers, orderer, and CA services

The setup shown in Figure 4.15 includes:

- Two organizations (`Org1` and `Org2`) each hosting one peer node.
- A single `Orderer` node responsible for block generation.
- A `Fabric CA` for issuing digital certificates.

Each peer container maintains its own copy of the ledger. When the backend submits a transaction via the Fabric Contract API, it is endorsed by both peers, ordered, and then committed to the blockchain.

In summary, the LEDGIS core services collectively uphold the system's guiding principles: security, transparency, and resilience. The backend manages logic and trust boundaries; the client provides a secure and user-friendly interface; IPFS guarantees distributed and immutable storage; and Hyperledger Fabric ensures tamper-proof record keeping. Together, they create a reliable digital infrastructure for storing, verifying, and retrieving legal evidence with confidence.

Chaincode Design

The chaincode, implemented in `Node.js` using the **fabric-contract-api**, defines two primary functions: `storeEvidence` and `getEvidence`. The former records encrypted evidence metadata into the blockchain ledger, while the latter retrieves stored records based on the evidence hash.

The code below represents the complete implementation used within the LEDGIS network.

```
’use strict’;
const { Contract } = require(’fabric-contract-api’);
class EvidenceContract extends Contract {
    async initLedger(ctx) {
        console.info(’Chaincode instantiated’);
    }
    async storeEvidence(ctx, evdString) {
        const {hash} = JSON.parse(evdString);
        await ctx.stub.putState(hash, Buffer.from(evdString));
        return {success:true,hash:hash};
    }
    async getEvidence(ctx,evID){
        const evBytes=await ctx.stub.getState(evID)
        if(!evBytes)return {msg:"invalid key",status:failed}
        return JSON.parse(evBytes.toString())
    }
}
module.exports = EvidenceContract;
```

Listing 4.4: EvidenceContract chaincode for storing and retrieving evidence metadata

The **storeEvidence()** method receives evidence details as a serialized JSON string, extracts the file hash, and commits it to the ledger using the Fabric **putState()** API. Each record is indexed by its hash, ensuring that evidence is uniquely identifiable and tamper-proof. The **getEvidence()** method allows retrieval of any stored metadata using the same hash key. Together, these methods provide a minimal yet effective on-chain storage mechanism for digital evidence metadata.

Backend Submission Controller

All communication with the blockchain occurs through the backend server, which acts as an intermediary between the client and the Hyperledger Fabric network. This design isolates blockchain complexity from the frontend while enforcing authentication and authorization through the backend layer.

The **submissioncontroller.js** file defines an asynchronous helper function **subToFabric()**, which connects to the Fabric network, submits chaincode transactions, and returns the result to the API endpoint.

```
const { Gateway, Wallets } = require('fabric-network');
const fs = require('fs');
const path = require('path');
const ccpPath = path.resolve(__dirname, '..','..', 'fabric-samples', 'test-network', 'organizations', 'peerOrganizations', 'org1.example.com', 'connection-org1.json');
const walletPath = path.join(__dirname, '..','wallet');
async function subToFabric(functionName, args) {
  try {
    const ccp = JSON.parse(fs.readFileSync(ccpPath, 'utf8'));
    const wallet = await Wallets.newFileSystemWallet(walletPath);
    const identity = await wallet.get('appUser');
    if (!identity) {
      return { error: 'No user identity' };
    }
    const gateway = new Gateway();
    await gateway.connect(ccp, {
      wallet,
      identity: 'appUser',
      discovery: { enabled: true, asLocalhost: true }
    });
    const network = await gateway.getNetwork('mychannel');
    const contract = network.getContract(’maincontract’);
    const result = await contract.submitTransaction(functionName, ...args);
    await gateway.disconnect();
    return { result: result.toString() };
  } catch (error) {
    console.error(‘Failed to submit transaction: ${error}‘);
    return { error: error.message };
  }
}
module.exports={subToFabric}
```

Listing 4.5: Backend submission controller for interacting with Hyperledger Fabric

In this implementation, the backend server first loads the connection profile (saved in `connection-org1.json`) and retrieves the registered user identity (`appUser`) from the local wallet. The connection profile contains details about the network's peers, orderers, and channels, while the wallet securely stores the user's certificates and private keys issued by the Fabric CA. These credentials allow the backend to authenticate and sign transactions, ensuring every blockchain operation is traceable and verified.

Once both configuration and identity are ready, a gateway connection is established to the `mychannel` network. The gateway simplifies peer discovery, endorsement, and transaction submission within Hyperledger Fabric. Through this connection, the backend accesses the deployed smart contract (`maincontract`) and invokes the required chaincode function using `submitTransaction(functionName, ...args)`. For example, during an evidence upload, it calls `storeEvidence` with the file hash and related metadata. The network endorses the transaction, commits it to the ledger, and returns the result, which the backend converts into a readable response.

After execution, the gateway connection is cleanly closed to free resources and maintain efficient operation. This modular setup allows the backend to handle both evidence submission and retrieval securely through API calls. The client never interacts with the blockchain directly; instead, all operations pass through the backend, which manages authentication, validation, and communication with the ledger. This design keeps the client lightweight while ensuring that every blockchain transaction—from upload to verification—remains secure, auditable, and tamper-resistant within the LEDGIS ecosystem.

CHAPTER 5

RESULTS AND DISCUSSION

This chapter presents a comprehensive evaluation of the LEDGIS framework, validating its functional correctness, quantitative performance, and practical applicability. The section begins by detailing the experimental environment, followed by a multi-faceted analysis of the results obtained, including functional test cases, quantitative performance benchmarks, and a comparative analysis against existing systems. The chapter concludes with a critical discussion of the insights gained, challenges faced, and limitations identified, ultimately proposing a clear roadmap for future development.

5.1 Experimentation Details

All functional and quantitative evaluations were conducted within a controlled testbed. This section meticulously details the hardware, software, and network architecture of this environment, which is essential for contextualizing and interpreting the performance results that follow.

5.1.1 Hardware Environment

The performance benchmarks were conducted on a modest, developer-grade hardware setup to establish a conservative baseline for system viability. The testbed comprised:

- **CPU:** Intel Core i3-7100U
- **RAM:** 16GB
- **Storage:** M.2 SSD

The use of non-server-grade hardware, particularly a dual-core, low-power processor, is a critical variable. The performance data (e.g., latency, throughput) derived from this testbed should be interpreted as a performance floor or a conservative baseline, not the system's maximum capability. Any performance bottlenecks identified, such as the saturation point observed under concurrent load, are partially attributable to this resource-constrained environment.

The system's ability to function effectively on this hardware strongly supports its feasibility, as performance would predictably and significantly improve with an enterprise-grade deployment.

5.1.2 Software and Network Architecture

The LEDGIS system is a multi-component, containerized application designed for modularity and scalability. The experimental setup employed a comprehensive software stack, beginning with the blockchain network built on Hyperledger Fabric v2.5, deployed in Docker containers. The network consisted of multiple peers, an orderer, and a Certificate Authority (CA), all operating within a controlled local environment. The CouchDB v3.2 database served as the state database for Fabric peers, enabling efficient storage and retrieval of world-state data.

For decentralized storage, a locally running IPFS daemon was integrated to handle evidence file storage and retrieval operations, ensuring data redundancy and persistence. The backend of the system was developed using Node.js (v18+) with the Express.js framework, serving as the central orchestrator that connected all components and interacted with the blockchain through the fabric-sdk-node interface. The frontend layer, implemented using Next.js, provided a seamless user interface for performing blockchain transactions, uploading files, and retrieving evidence records. All these components — Fabric, IPFS, backend, and frontend — were containerized using Docker and interconnected via a dedicated Docker bridge network to ensure isolation and consistent inter-service communication.

This microservice-based architecture implies that the measured API latency does not represent solely the "blockchain speed." Instead, it reflects the cumulative time required for a series of interdependent service calls.

For example, a single `POST /upload` operation involves several sequential processes — an initial HTTP request, local cryptographic operations, interaction with the IPFS daemon, and a complete transaction submission to the Fabric peer. This inherent "orchestration overhead" contributes significantly to the total measured latency, suggesting that the core components such as Fabric and IPFS are, in fact, more efficient than the aggregate timings might initially indicate.

5.2 Results Obtained

This section presents the results of the project, showing that the system works correctly (the "what") and performs efficiently (the "how well"). The snapshots and test cases from the original report are included here as they provide key evidence supporting these results.

5.2.1 Functional Validation and test case validation

This section provides qualitative and empirical evidence that all critical system components and pipelines function as designed. Validation was performed at both the API level using **Postman** and the application level via **UI snapshots**.

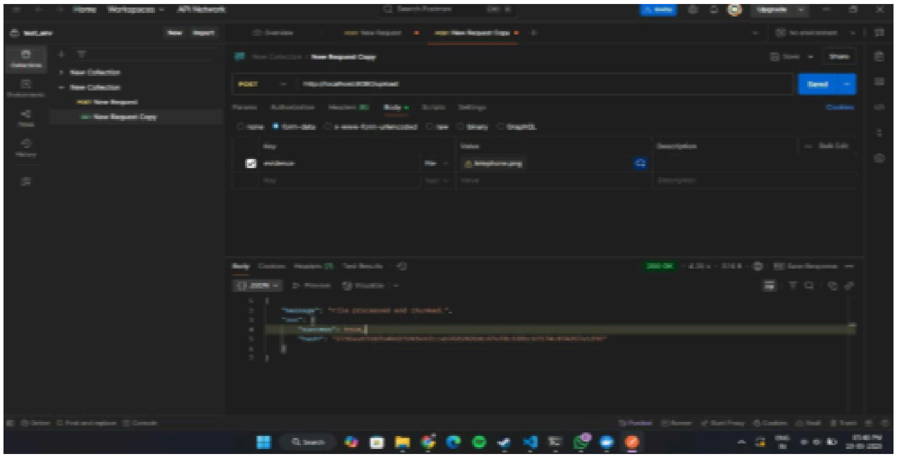
API-Level Test Case: User Authentication (RBAC)

The system's Role-Based Access Control (RBAC) mechanism was validated through a series of API tests. A successful request to the `/login` endpoint correctly authenticated a valid user and returned a JSON Web Token (JWT). A subsequent test involved accessing an admin-only route using this JWT in the authorization header. The backend middleware correctly validated the token and authorized access based on the "Admin" role. This confirms the correct implementation of the system's security and RBAC model.

API-Level Test Case: End-to-End Evidence Upload Pipeline

A `POST` request was sent to the evidence upload API endpoint with a sample file. The test returned a 200 OK status and a JSON response containing the file's unique hash (generated using SHA-256) and the IPFS Content Identifiers (CIDs) for the stored chunks. This JSON response serves as direct proof of the successful execution of the entire Evidence Upload and Secure Storage Pipeline.

A successful response indicates that all critical steps—(1) hashing, (2) AES encryption, (3) chunking, (4) upload to IPFS, (5) metadata creation, and (6) storing metadata on the ledger—completed successfully. Any failure in this sequence would have triggered an error, thereby confirming the integrity and robustness of the cryptographic and distributed storage pipeline.



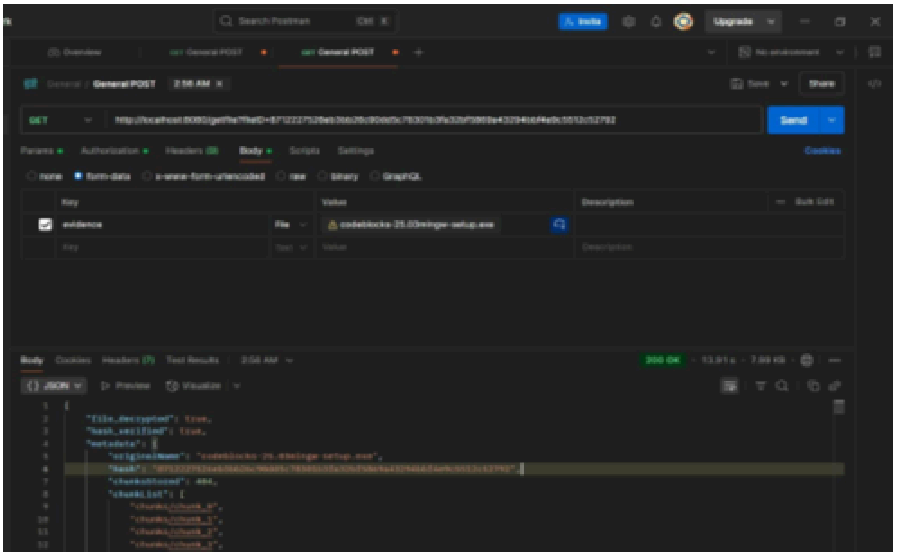
POST http://localhost:8080/upload

Key	Value	Description
evidence	telephone.png	

200 OK

```
{
    "message": "File processed and chunked.",
    "[illegible]": {
        "success": true,
        "hash": "[illegible]"
    }
}
```

Figure 5.1: Postman Test: Evidence Upload API returning success with hash and CID storage confirmation



GET http://localhost:8080/getFile?fileID=[illegible]

Key	Value	Description
evidence	codeblocks-25.03mingw-setup.exe	

200 OK

```
{
    "file_decrypted": true,
    "hash_verified": true,
    "metadata": {
        "originalName": "codeblocks-25.03mingw-setup.exe",
        "hash": "[illegible]",
        "chunksStored": 484,
        "chunkList": [
            "chunks/chunk_0",
            "chunks/chunk_1",
            "chunks/chunk_2",
            "chunks/chunk_3",
```

Figure 5.2: Postman Test: Metadata Retrieval showing hash, key, IV, and chunk references from ledger

API-Level Test Case: End-to-End Evidence Retrieval Pipeline

A `GET` request was sent to the metadata retrieval endpoint using a known file hash as a key. The system successfully queried the Hyperledger Fabric ledger through the `getEvidence` chaincode function and returned the stored JSON metadata object. The object contained the file's hash, AES key, IV, and chunk references, validating the first half of the Evidence Retrieval and Verification Pipeline.

This test confirms that metadata stored on-chain remains persistent, immutable, and retrievable. Furthermore, the retrieved key, IV, and CIDs serve as inputs for subsequent decryption and integrity verification processes.

Application-Level (Snapshot) Validation

Snapshots of the client application confirm that the user interface functions as intended. The Evidence Commit Page provides an intuitive interface for initiating the upload process . Additionally, the IPFS Map visualization illustrates active peers and storage statistics, demonstrating that the local IPFS service is successfully connected to the global peer-to-peer network rather than operating in isolation.

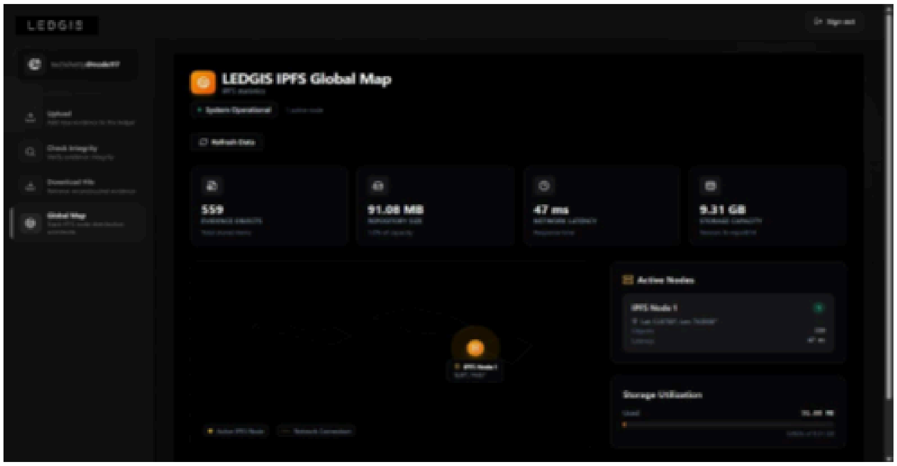


Figure 5.3: Global IPFS Map: A webpage showcasing the global ipfs storage system nodes along with the node statistics

5.2.2 Performance Analysis for Scalability

This section analyzes the system's performance in terms of file processing and the overhead involved in data ingestion. Quantitative data has been drawn from the research

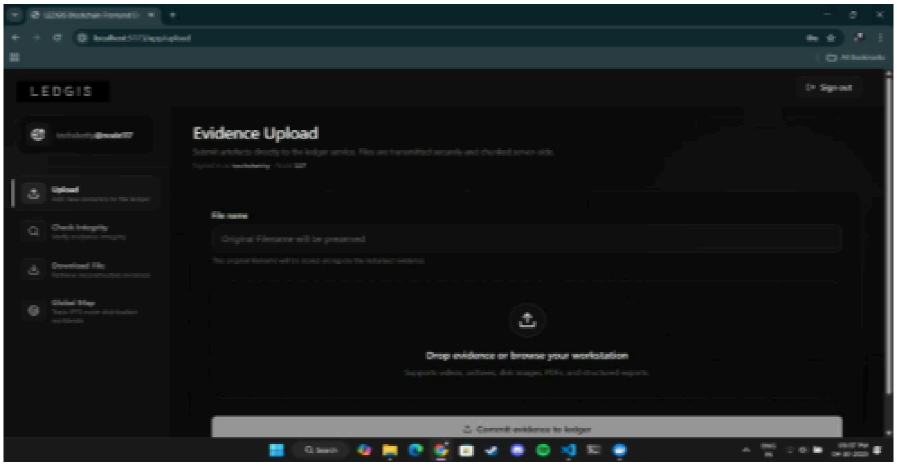

LEDGIS
Sign out
Upload
Check Integrity
Download File
Global Map
Evidence Upload
File name
Original Filename will be preserved
Drop evidence or browse your workstation
Commit evidence to ledger

Figure 5.4: Evidence Commit Page: Interface to submit new evidence to the ledger

phase of the project. The system demonstrates highly efficient and nearly linear scaling for local file operations such as hashing, encryption, and chunking. As observed in the performance benchmark, a five-megabyte file is processed almost instantly, while a one-hundred-sixty-megabyte file is completely processed in about 1.2 seconds. This behavior indicates a linear scalability trend, confirming that local cryptographic operations are not a system bottleneck, even for large evidence files. These tasks are CPU-bound and scale predictably with input size, which validates the system's computational efficiency.

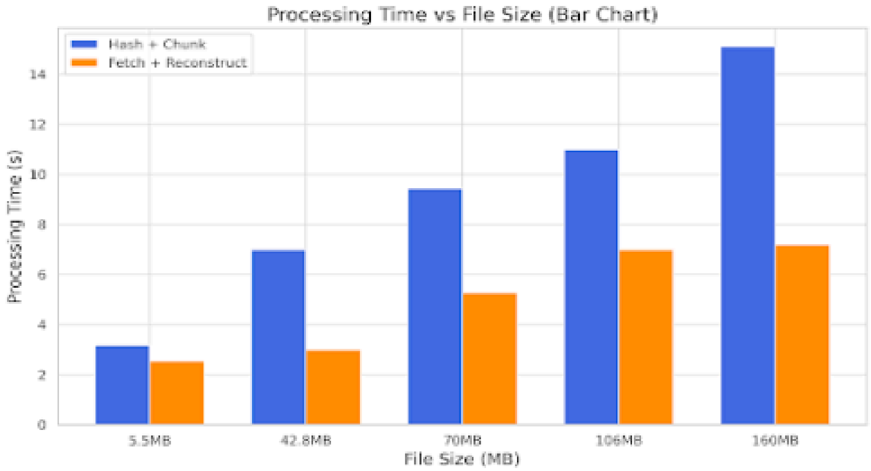


Figure 5.5: System processing time for handling files of various sizes, demonstrating a linear performance scaling that supports large files

A clear difference in latency was observed between data ingestion and retrieval operations. The average latency for the POST upload operation was approximately 10,489 milliseconds, whereas the average latency for the GET retrieval operation was around

3,449 milliseconds. This difference highlights the expected performance characteristics of the hybrid architecture. Local processing of even large files is completed in about 1.2 seconds, while the total upload duration of roughly 10.5 seconds accounts for the additional overhead of uploading encrypted chunks to the distributed IPFS network and submitting metadata transactions to the Hyperledger Fabric network for consensus and block commitment.

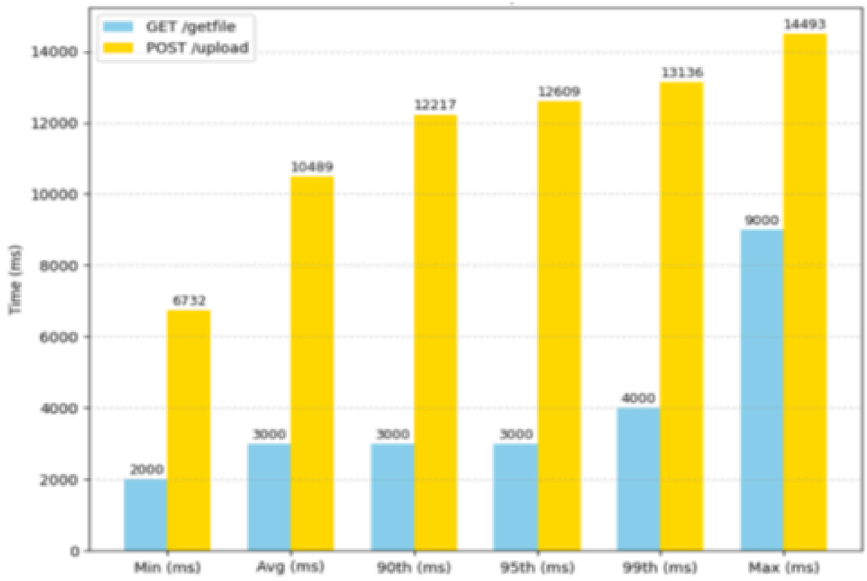


Figure 5.6: A comparison of API response times, showing the higher latency of the ingestion process due to cryptographic and consensus overhead.

This ten-second duration represents the deliberate cost of achieving immutability and trust through blockchain consensus. Attempting to store the entire file directly on-chain would lead to unmanageable block sizes, failed transaction propagation, and an unusable ledger. By isolating only the lightweight metadata—such as the hash, CIDs, and encryption key—on the blockchain, while storing the actual file data on IPFS, the system effectively separates time-intensive trust operations from fast local processing. This approach confirms that the hybrid architecture provides the right balance between scalability, integrity, and performance.

5.3 Societal Impact of the Project

The implementation of the LEDGIS framework extends beyond immediate technical utility, offering significant tangible benefits to the broader legal and governmental ecosystem.

By transitioning from vulnerable centralized databases to a tamper-proof decentralized ledger, the project addresses several critical societal needs:

Restoration of Public Trust in Justice: The platform directly addresses the erosion of confidence in public institutions caused by data manipulation and lack of transparency. By ensuring that legal records and evidence are immutable and mathematically verifiable, the system guarantees that judicial outcomes are based on authentic, unaltered facts, thereby strengthening the social contract between the state and its citizens.

Enhancement of Institutional Accountability: The integration of immutable audit trails creates a culture of transparency where every interaction with sensitive data is logged and traceable. This verifiable history acts as a deterrent against corruption and unauthorized data tampering, ensuring that government officials and law enforcement agencies remain accountable for their handling of public records.

Acceleration of Legal Procedures: The system addresses the societal issue of ”justice delayed is justice denied” by reducing bureaucratic friction. By automating document verification and evidence chain-of-custody through smart contracts, the framework minimizes administrative delays and human errors, expediting legal workflows and helping to reduce the backlog of pending court cases.

Protection of Civil Liberties: The shift to a permissioned blockchain architecture with robust encryption ensures that sensitive citizen data is protected against cyberattacks and unauthorized leaks. By enforcing strict role-based access control, the system protects the privacy rights of individuals involved in legal proceedings while ensuring that law enforcement has the necessary tools to uphold public safety.

5.4 SDG (Sustainable Development Goals) Mapped

The LEDGIS framework is designed to align with the United Nations Sustainable Development Goals (SDGs), focusing on strengthening institutions, fostering innovation, and ensuring social justice. Out of the 17 defined goals, our project specifically targets and contributes to the following areas through its decentralized architecture and security protocols:

- **Goal 16: Peace, Justice, and Strong Institutions (Social/Governance Category):** This project offers a blockchain-based solution for the secure management and auditability of legal and law enforcement records. It directly contributes to Goal 16 by promoting the rule of law and ensuring equal access to justice. By utilizing Hyperledger Fabric to prevent data manipulation, the system fosters effective, accountable, and transparent institutions at all levels.

- **Goal 9: Industry, Innovation, and Infrastructure (Economic Category):** The implementation of a hybrid on-chain and off-chain architecture (using IPFS and Hyperledger Fabric) contributes to Goal 9 by upgrading the technological infrastructure of government sectors. The project introduces a resilient, decentralized alternative to traditional, vulnerable centralized databases, thereby fostering innovation in public service delivery and enhancing digital security standards.
- **Goal 10: Reduced Inequalities (Social Category):** By guaranteeing the integrity of digital evidence through cryptographic hashing and immutable logging, LEDGIS contributes to Goal 10. It helps eliminate systemic loopholes that allow for evidence tampering, ensuring that judicial outcomes are determined by authentic facts rather than external influence. This technological impartiality is crucial for reducing inequalities in the application of justice.

Table 5.1: Mapping Project Features to Relevant SDGs

Target SDG	**Project Module**	**Contribution to Goal**
SDG 16: Peace, Justice & Strong Institutions	Immutable Ledger	Ensures transparency and accountability by creating tamper-proof audit trails for all legal records.
SDG 9: Industry, Innovation & Infrastructure	Hybrid Architecture	Modernizes government IT infrastructure with scalable, secure, and resilient blockchain technology.
SDG 10: Reduced Inequalities	Smart Contracts	Automates access control to ensure equal and unbiased application of evidence handling procedures.

SUSTAINABLE DEVELOPMENT GOALS
1 NO POVERTY
2 ZERO HUNGER
3 GOOD HEALTH AND WELL-BEING
4 QUALITY EDUCATION
5 GENDER EQUALITY
6 CLEAN WATER AND SANITATION
7 AFFORDABLE AND CLEAN ENERGY
8 DECENT WORK AND ECONOMIC GROWTH
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE
10 REDUCED INEQUALITIES
11 SUSTAINABLE CITIES AND COMMUNITIES
12 RESPONSIBLE CONSUMPTION AND PRODUCTION
13 CLIMATE ACTION
14 LIFE BELOW WATER
15 LIFE ON LAND
16 PEACE, JUSTICE AND STRONG INSTITUTIONS
17 PARTNERSHIPS FOR THE GOALS
SUSTAINABLE DEVELOPMENT GOALS

Figure 5.7: Sustainable Development Goals (SDGs)

5.5 Discussions

This section interprets the findings of the project and discusses their broader implications, challenges, and identified limitations.

5.5.1 Insights from the Study

The study demonstrates that the proposed system achieves verifiable immutability of digital evidence. The upload process effectively acts as a digital notary, creating an immutable, time-stamped record of an evidence hash on the ledger. The retrieval pipeline then functions as the auditor, verifying the integrity of the evidence against this recorded hash each time it is accessed. Together, these processes establish a closed-loop, cryptographically secured chain of custody essential for judicial reliability.

Another key insight is the automation of the chain-of-custody process and the shift of trust from human oversight to automated, verifiable protocols. Historically, evidence handling relied heavily on manual logs and subjective verification, which were prone to human error. By using smart contracts and automated backend workflows, the framework eliminates human intervention in evidence logging and verification. This transition from human trust to protocol-based verification provides significant governance and efficiency benefits in legal and investigative workflows.

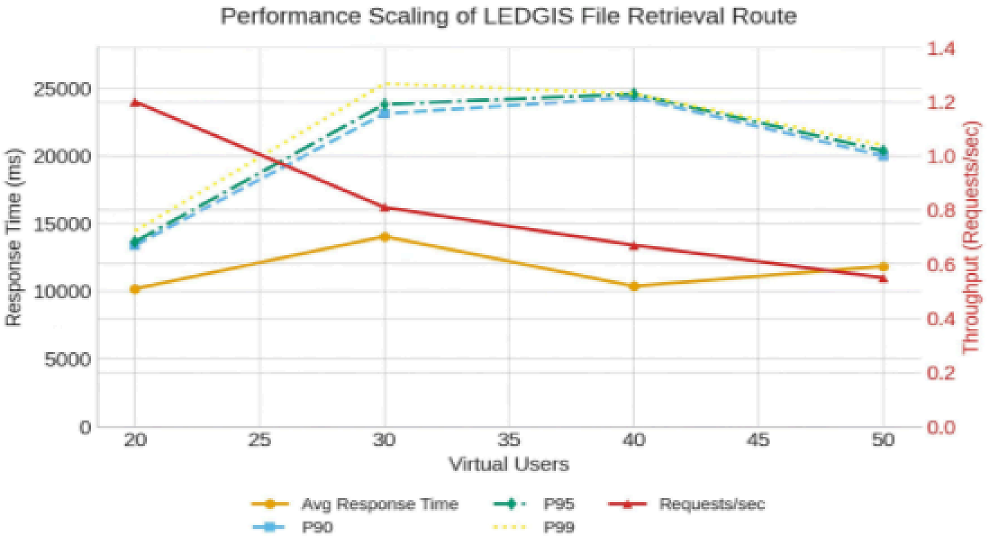


Figure 5.8: System performance under concurrent load, illustrating the relationship between throughput and response time as user load increases

5.5.2 Challenges and Problems Faced

The main technical difficulty encountered during implementation was the orchestration of multiple interdependent services, including the frontend, backend, blockchain network, state database, and decentralized storage. Managing data flow, handling cryptographic identities such as the application wallet, and ensuring proper error propagation across these containerized components required careful configuration and debugging.

Another important challenge involved cryptographic key management. A practical decision was made in the prototype phase to store AES encryption keys and initialization vectors on-chain along with metadata. While this approach facilitated testing and validation, it is not considered secure for a production environment. This limitation emphasizes the need for a robust, external key management mechanism in future iterations to enhance overall system security and compliance.

5.6 Conclusion of results

This chapter presented a detailed validation of the LEDGIS framework. The functional tests confirmed that the cryptographic and distributed storage pipelines work correctly from end to end. The performance analysis verified the efficiency of the hybrid on-chain and off-chain design, showing that it can handle large files with predictable, linear scaling and reasonable upload times. Load testing established a performance baseline and identified the system's limits, providing useful information for future improvements. The comparison study highlighted LEDGIS as a unique, domain-specific solution for legal and forensic chain-of-custody management. Finally, this chapter discussed key findings, technical challenges, and limitations such as prototype-level key management, outlining clear directions for future research and system enhancement.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

This final chapter summarizes the overall findings of the project, evaluates how well the objectives were achieved, and presents a practical roadmap for future improvements. It also reflects on the lessons learned during the design and implementation of the LEDGIS framework and outlines ways to make the system more secure, efficient, and suitable for real-world use.

6.1.1 Summary of Findings

This project aimed to address the challenge of securing government and law enforcement records by developing a blockchain-based framework named LEDGIS. Through the design and implementation of this system, the project demonstrated that combining a permissioned blockchain (Hyperledger Fabric) for metadata and consensus with decentralized off-chain storage (IPFS) for encrypted data provides a secure and scalable solution.

Two main findings were established. First, the system ensures strong security and data integrity. It maintains a tamper-proof ledger of all evidence metadata, creating a verifiable and auditable chain of custody for legal and forensic use. Second, the hybrid architecture successfully solves the scalability problem that affects many blockchain applications. Performance tests showed that the system scales in a predictable and linear manner with file size, proving that it can handle large digital evidence files without being limited by blockchain transaction overhead.

6.1.2 Significance and Contribution of the LEDGIS Framework

The project demonstrates that blockchain technology can be applied in a meaningful and practical way to improve digital governance and evidence management. The main contribution of this work is a working prototype that connects theoretical blockchain ideas with the practical requirements of real-world government systems. By validating the hybrid on-chain and off-chain approach, the project provides a clear architectural model that balances cryptographic security with operational scalability. This fills an important gap that was identified in many existing studies, where systems were often either secure but slow, or scalable but less verifiable.

6.1.3 Final Remarks on Project Objectives

All the main objectives outlined at the beginning of the project were successfully achieved. A Hyperledger Fabric-based system was implemented to securely manage digital records. A hybrid processing pipeline was developed to hash and encrypt files, store them off-chain, and verify them through blockchain metadata. Smart contracts were written to automate the registration and validation of digital evidence. Role-based access control was integrated at the backend level, ensuring only authorized users could access or modify records. Finally, extensive testing and performance benchmarking were conducted to analyze scalability and identify limitations, which now form the foundation for future improvements.

6.2 Future Enhancements

The current LEDGIS system functions as a working proof-of-concept. However, several areas have been identified for improvement to make it production-ready and capable of supporting real-world deployment. The following subsections describe key directions for future development.

6.2.1 Architectural Enhancements: On-Chain Access Control

One of the main limitations of the current prototype is that access control is handled by a centralized backend, which introduces a single point of trust. A future version of LEDGIS should move this logic to the blockchain itself by implementing an Attribute-Based Access Control (ABAC) model within the chaincode.

In this approach, the Hyperledger Fabric Certificate Authority would issue digital certificates containing user-specific attributes such as role, department, and clearance level. The smart contract functions, such as getEvidence, would then verify these attributes directly before allowing access to sensitive records. New chaincode functions like setAccessPolicy could allow administrators to define and store fine-grained access policies on the ledger, ensuring that permissions are enforced transparently and immutably. This change would eliminate the need to trust a central server and make access control entirely decentralized.

6.2.2 Performance and Scalability Improvements

Performance testing revealed some bottlenecks in the current system, especially in reading and writing large files. To improve scalability, the backend server can be deployed in a containerized environment and scaled horizontally using a load balancer or Kubernetes cluster. This would distribute decryption and file reconstruction tasks across multiple nodes, improving throughput during evidence retrieval.

For write operations, additional testing should be conducted to measure performance under heavy concurrent uploads. The system's transaction throughput can be improved by tuning Hyperledger Fabric's block creation parameters, such as BatchTimeout and MaxMessageCount, and by batching multiple metadata transactions into a single block. To reduce latency, a caching layer such as Redis could be introduced to temporarily store frequently accessed metadata, improving response times for end users.

6.2.3 Advanced Cryptography and Privacy Preservation

The current system stores AES encryption keys in plaintext form on the blockchain, which poses a potential security risk. Future versions should replace this with a more advanced encryption method such as Proxy Re-Encryption (PRE). This technique allows encrypted files to be securely shared between authorized users without ever exposing the actual encryption key.

For example, a police officer could encrypt a file with a secret key and then encrypt that key with their public key before storing it on the ledger. When a judge needs access, a special re-encryption key can be used to transform the encrypted key so that only the judge can decrypt it. At no point is the actual encryption key exposed to the network. This approach allows secure and revocable access control, ensuring privacy while maintaining auditability. In future, searchable encryption could also be integrated to allow users to search encrypted data without revealing their search terms.

6.2.4 System Interoperability and Governance

For the system to be useful in real-world government settings, it must be capable of integrating with other platforms. Future work should focus on developing a standardized API that enables LEDGIS to connect with existing systems such as digital court records, police databases, or e-FIR portals. To support inter-blockchain communication, frameworks like Hyperledger Cactus could be used to exchange verification data with other government blockchain networks, such as those managing land or identity records.

Additionally, governance mechanisms should be introduced to manage the network itself. A separate governance chaincode could handle the process of adding new organizations or courthouses to the network, managing membership policies, and maintaining version control for system updates. This would make the system self-regulating and suitable for deployment across multiple agencies.

REFERENCES

[1] S. Liu and Q. Zheng, "A study of a blockchain-based judicial evidence preservation scheme," *Blockchain: Research and Applications*, vol. 5, no. 2, p. 100192, 2024.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," May 2009.

[3] L. Foundation, "Hyperledger blockchain frameworks for business and government." `https://www.hyperledger.org`.

[4] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *2017 IEEE International Congress on Big Data (BigData Congress)*, p. 557–564, IEEE, June 2017.

[5] D. Tapscott and A. Tapscott, *Blockchain Revolution: How the Technology Behind Bitcoin Is Changing Money, Business, and the World*. Penguin Publishing Group, 2016.

[6] G. of Estonia, "Estonia's digital government and blockchain implementation." `https://e-estonia.com`.

[7] G. Ministry of Justice, "Blockchain for land registry: A case study on secure government records." `https://gov.ge/blockchain-land-registry`.

[8] M. Risius and K. Spohrer, "A blockchain research framework: What we (don't) know, where we go from here, and how we will get there," *Business &amp; Information Systems Engineering*, vol. 59, p. 385–409, Dec. 2017.

[9] M. Atzori, "Blockchain technology and decentralized governance: Is the state still necessary?," *Journal of Governance and Regulation*, vol. 6, no. 1, p. 45–62, 2017.

[10] N. Kshetri, "Blockchain's roles in strengthening cybersecurity and protecting privacy," *Telecommunications Policy*, vol. 41, p. 1027–1038, Nov. 2017.

[11] D. Yaga, P. Mell, N. Roby, and K. Scarfone, *Blockchain technology overview.* Oct. 2018.

[12] M. Swan, *Blockchain: Blueprint for a New Economy.* O'Reilly, 2015.

[13] J. Sun, J. Yan, and K. Z. K. Zhang, "Blockchain-based sharing services: What blockchain technology can contribute to smart cities," *Financial Innovation*, vol. 2, Dec. 2016.

[14] M. Pilkington, *Blockchain technology: principles and applications.* Edward Elgar Publishing, Sept. 2016.

[15] E. Commission, "Blockchain for digital government: A european perspective." `https://ec.europa.eu/digital-strategy`, 2019.

[16] L. W. Cong and Z. He, "Blockchain disruption and smart contracts," *The Review of Financial Studies*, vol. 32, p. 1754–1797, Apr. 2019.

[17] G. Zyskind, O. Nathan, and A. S. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," in *2015 IEEE Security and Privacy Workshops*, p. 180–184, IEEE, May 2015.

[18] N. D. Bhaskar and D. L. K. Chuen, *Bitcoin Exchanges*, p. 559–573. Elsevier, 2015.

[19] K. Saito and H. Yamada, "What's so different about blockchain? – blockchain is a probabilistic state machine," in *2016 IEEE 36th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, p. 168–175, IEEE, June 2016.

[20] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare data gateways: Found healthcare intelligence on blockchain with novel privacy risk control," *Journal of Medical Systems*, vol. 40, Aug. 2016.

[21] M. Kassen, "Blockchain and public service delivery: a lifetime cross-referenced model for e-government," *Enterprise Information Systems*, vol. 18, Feb. 2024.

[22] A. A. Mamun, S. Azam, and C. Gritti, "Blockchain-based electronic health records management: A comprehensive review and future research direction," *IEEE Access*, vol. 10, p. 5768–5789, 2022.

[23] M. J. Sousa, "Blockchain as a driver for transformations in the public sector," *Policy Design and Practice*, vol. 6, p. 415–432, Oct. 2023.

[24] S. Khan, M. Shael, M. Majdalawieh, N. Nizamuddin, and M. Nicho, “Blockchain for governments: The case of the dubai government,” *Sustainability*, vol. 14, p. 6576, May 2022.

[25] E. Tan, S. Mahula, and J. Crompvoets, “Blockchain governance in the public sector: A conceptual framework for public management,” *Government Information Quarterly*, vol. 39, p. 101625, Jan. 2022.

[26] T.-H. Kim, G. Kumar, R. Saha, M. K. Rai, W. J. Buchanan, R. Thomas, and M. Alazab, “A privacy preserving distributed ledger framework for global human resource record management: The blockchain aspect,” *IEEE Access*, vol. 8, p. 96455–96467, 2020.

[27] C. Piao, Y. Hao, J. Yan, and X. Jiang, “Privacy preserving in blockchain-based government data sharing: A service-on-chain (soc) approach,” *Information Processing &amp; Management*, vol. 58, p. 102651, Sept. 2021.

[28] N. Elisa, L. Yang, F. Chao, N. Naik, and T. Boongoen, “A secure and privacy-preserving e-government framework using blockchain and artificial immunity,” *IEEE Access*, vol. 11, p. 8773–8789, 2023.

[29] J. Mahlaba, A. K. Mishra, D. Puthal, and P. K. Sharma, “Blockchain-based sensitive document storage to mitigate corruptions,” *IEEE Transactions on Engineering Management*, vol. 71, p. 12635–12647, 2024.

[30] F. Wang, Y. Gai, and H. Zhang, “Blockchain user digital identity big data and information security process protection based on network trust,” *Journal of King Saud University - Computer and Information Sciences*, vol. 36, p. 102031, Apr. 2024.

[31] L. Chen, W.-K. Lee, C.-C. Chang, K.-K. R. Choo, and N. Zhang, “Blockchain based searchable encryption for electronic health record sharing,” *Future Generation Computer Systems*, vol. 95, p. 420–429, June 2019.

[32] X. Tang, C. Guo, K.-K. R. Choo, Y. Liu, and L. Li, “A secure and trustworthy medical record sharing scheme based on searchable encryption and blockchain,” *Computer Networks*, vol. 200, p. 108540, Dec. 2021.

[33] J. Liu, X. Li, L. Ye, H. Zhang, X. Du, and M. Guizani, "Bpds: A blockchain based privacy-preserving data sharing for electronic medical records," in *2018 IEEE Global Communications Conference (GLOBECOM)*, p. 1–6, IEEE, Dec. 2018.

[34] H. Li and D. Han, "Edurss: A blockchain-based educational records secure storage and sharing scheme," *IEEE Access*, vol. 7, p. 179273–179289, 2019.

[35] M. Naz, F. A. Al-zahrani, R. Khalid, N. Javaid, A. M. Qamar, M. K. Afzal, and M. Shafiq, "A secure data sharing platform using blockchain and interplanetary file system," *Sustainability*, vol. 11, p. 7054, Dec. 2019.

[36] Z. Ullah, B. Raza, H. Shah, S. Khan, and A. Waheed, "Towards blockchain-based secure storage and trusted data sharing scheme for iot environment," *IEEE Access*, vol. 10, p. 36978–36994, 2022.

[37] R. G. Sonkamble, A. M. Bongale, S. Phansalkar, A. Sharma, and S. Rajput, "Secure data transmission of electronic health records using blockchain technology," *Electronics*, vol. 12, p. 1015, Feb. 2023.

[38] S. Vidhya and V. Kalaivani, "A blockchain based secure and privacy aware medical data sharing using smart contract and encryption scheme," *Peer-to-Peer Networking and Applications*, vol. 16, p. 900–913, Jan. 2023.

[39] G. Verma and S. Kanrar, "Secure document sharing model based on blockchain technology and attribute-based encryption," *Multimedia Tools and Applications*, vol. 83, p. 16377–16394, July 2023.

[40] W. Su and L. Ma, "A blockchain-based covert document communication system model," in *2023 8th International Conference on Computer and Communication Systems (ICCCS)*, p. 445–450, IEEE, Apr. 2023.

[41] S. Pandey, V. Rishiwal, D. S. Jat, P. Yadav, M. Yadav, and A. Jain, "Towards securing the digital document using blockchain technology with off-chain attribute based encryption framework," in *2024 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC)*, p. 857–864, IEEE, July 2024.

[42] S. B. C, R. Jacob, and B. Sowmiya, "Security and scalability of blockchain document storage systems," in *2024 8th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, p. 586–591, IEEE, Oct. 2024.

[43] M. Alruwaill, S. Mohanty, and E. Kougianos, “hchain: Blockchain based large scale ehr data sharing with enhanced security and privacy,” 2025.

[44] A. Siva Kumar, S. Godfrey Winster, and R. Ramesh, “Efficient sensitivity orient blockchain encryption for improved data security in cloud,” *Concurrent Engineering*, vol. 29, p. 249–257, Apr. 2021.

[45] Z. Lejun, P. Minghui, W. Weizheng, S. Yansen, C. Shuna, and K. Seokhoon, “Secure and efficient medical data storage and sharing scheme based on double blockchain,” *Computers, Materials &amp; Continua*, vol. 66, no. 1, p. 499–515, 2020.

[46] M. Kandpal, Y. Pritwani, C. Misra, A. Yadav, and R. Barik, “Towards data storage scheme in blockchain based serverless environment: Aes encryption and decryption algorithm approach,” *Facta universitatis - series: Electronics and Energetics*, vol. 37, no. 2, p. 317–342, 2024.

[47] S. Zafar, S. F. U. Hassan, A. Mohammad, A. A. Al-Ahmadi, and N. Ullah, “Implementation of a distributed framework for permissioned blockchain-based secure automotive supply chain management,” *Sensors*, vol. 22, p. 7367, Sept. 2022.

[48] S. Kushch, Y. Baryshev, and S. Ranise, “Blockchain tree as solution for distributed storage of personal id data and document access control,” *Sensors*, vol. 20, p. 3621, June 2020.

[49] S. S. N and M. Yadlapalli, “Block chain-based secure document sharing,” *Indian Journal of Computer Science and Technology*, p. 204–208, Apr. 2025.

[50] Y. Chen, S. Ding, Z. Xu, H. Zheng, and S. Yang, “Blockchain-based medical records secure storage and medical service framework,” *Journal of Medical Systems*, vol. 43, Nov. 2018.

[51] K. Fan, S. Wang, Y. Ren, H. Li, and Y. Yang, “Medblock: Efficient and secure medical data sharing via blockchain,” *Journal of Medical Systems*, vol. 42, June 2018.

[52] A. Dubovitskaya, Z. Xu, S. Ryu, M. Schumacher, and F. Wang, “Secure and trustable electronic medical records sharing using blockchain,” 2017.

[53] S. Wu and J. Du, "Electronic medical record security sharing model based on blockchain," in *Proceedings of the 3rd International Conference on Cryptography, Security and Privacy*, ICCSP 2019, p. 13–17, ACM, Jan. 2019.

[54] I. M. Akbar, A. Bhawiyuga, and R. Siregar, "An ethereum blockchain based electronic health record system for inter-hospital secure data sharing," in *6th International Conference on Sustainable Information Engineering and Technology 2021*, SIET '21, p. 226–230, ACM, Sept. 2021.

[55] V. B, S. N. Dass, S. R, and R. Chinnaiyan, "A blockchain based electronic medical health records framework using smart contracts," in *2021 International Conference on Computer Communication and Informatics (ICCCI)*, p. 1–4, IEEE, Jan. 2021.

[56] N. Nautiyal, P. Agarwal, and S. Sharma, "Rechain: A secured blockchain-based digital medical health record management system," in *2023 4th International Conference on Innovative Trends in Information Technology (ICITIIT)*, p. 1–6, IEEE, Feb. 2023.

[57] H. Wang, "Que bian: An electronic medical record management system on blockchain," in *2020 the 3rd International Conference on Blockchain Technology and Applications*, ICBTA 2020, p. 47–49, ACM, Dec. 2020.

[58] H. S. Musa, M. Krichen, A. A. Altun, and M. Ammi, "Survey on blockchain-based data storage security for android mobile applications," *Sensors*, vol. 23, p. 8749, Oct. 2023.

[59] A. R. Rajput, Q. Li, and M. T. Ahvanooey, "A blockchain-based secret-data sharing framework for personal health records in emergency condition," *Healthcare*, vol. 9, p. 206, Feb. 2021.

[60] W. Peng, T. Lu, W. Peng, and Z. Wang, "An efficient blockchain-based framework for file sharing," *Scientific Reports*, vol. 14, Aug. 2024.

[61] I.Bhuvaneshwarri and M. N. Sudha, "An implementation of secure storage using blockchain technology on cloud environment," *The Scientific Temper*, vol. 14, p. 806–810, Sept. 2023.

[62] S. Das, M. Mishra, R. Priyadarshini, R. K. Barik, and M. J. Saikia, "A secure, privacy-preserving, and cost-efficient decentralized cloud storage framework using

blockchain," *Journal of King Saud University - Computer and Information Sciences*, vol. 36, p. 102260, Dec. 2024.

[63] M. Almasian and A. Shafieinejad, "Secure cloud file sharing scheme using blockchain and attribute-based encryption," *Computer Standards &amp; Interfaces*, vol. 87, p. 103745, Jan. 2024.

[64] M. A. Al-Khasawneh, M. Faheem, A. A. Alarood, S. Habibullah, and A. Alzahrani, "A secure blockchain framework for healthcare records management systems," *Healthcare Technology Letters*, vol. 11, p. 461–470, Oct. 2024.

[65] A. Punia, P. Gulia, N. S. Gill, E. Ibeke, C. Iwendi, and P. K. Shukla, "A systematic review on blockchain-based access control systems in cloud environment," *Journal of Cloud Computing*, vol. 13, Sept. 2024.

[66] O. Dalvi, H. Javkar, K. M. Zaid, and M. M. Gedam, "Blockcert: Blockchain based document verification system," *International Research Journal on Advanced Engineering Hub (IRJAEH)*, vol. 3, p. 1736–1742, Apr. 2025.

[67] M. S. Rahman, I. Khalil, P. C. Mahawaga Arachchige, A. Bouras, and X. Yi, "A novel architecture for tamper proof electronic health record management system using blockchain wrapper," in *Proceedings of the 2019 ACM International Symposium on Blockchain and Secure Critical Infrastructure*, Asia CCS '19, p. 97–105, ACM, July 2019.

[68] X. Luo, X. Chen, X. Chen, and Q. Cheng, "A survey on the application of blockchain in cryptographic protocols," *Cybersecurity*, vol. 7, Dec. 2024.

[69] W. Huang, X. Yu, and Z. Ma, "A study on blockchain-based data proxy re-encryption privacy protection," in *Proceedings of the 2024 3rd International Conference on Cryptography, Network Security and Communication Technology*, CNSCT 2024, p. 25–29, ACM, Jan. 2024.

[70] Q. Li and Q. Zhou, "Design of blockchain traceability mechanism for data privacy protection," in *Proceedings of the 2024 2nd International Conference on Internet of Things and Cloud Computing Technology*, IoTCCT 2024, p. 312–316, ACM, Sept. 2024.

[71] Z. Ying and K. Wang, "Blockchain distributed identity management model for cross-border data privacy protection," *Journal of Surveillance, Security and Safety*, vol. 4, p. 112–28, Dec. 2023.

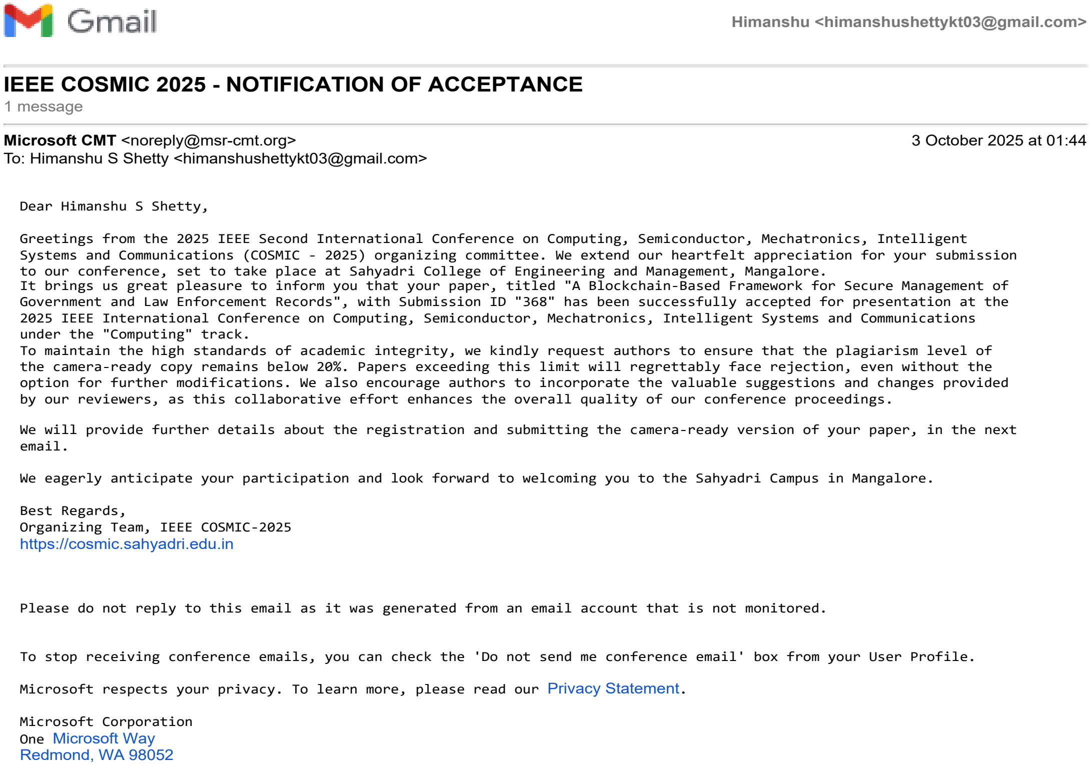
[72] Y. Zhang, P. Zhang, M. Guizani, J. Zhang, J. Wang, H. Zhu, K. K. Igorevich, and H. Shi, "Blockchain-based secure communication of internet of things in space–air–ground integrated network," *Future Generation Computer Systems*, vol. 158, p. 391–399, Sept. 2024.

[73] R. D. Garcia, G. Ramachandran, K. Dunnett, R. Jurdak, C. Ranieri, B. Krishnamachari, and J. Ueyama, "A survey of blockchain-based privacy applications: An analysis of consent management and self-sovereign identity approaches," 2024.

[74] X. Wang, W. Xiao, N. Liu, K. Xiao, Z. Gao, Y. Yang, and Y. Wang, *Research on the Mechanism of Privacy-Enhanced Cross-Institutional Data Sharing*, p. 29–40. Springer Nature Singapore, Oct. 2025.

APPENDIX - A : PLAGIARISM REPORT FRONT PAGE (PROJECT REPORT)

APPENDIX - B : PAPER PUBLICATION DETAILS



Gmail

Himanshu <himanshushettykt03@gmail.com>

IEEE COSMIC 2025 - NOTIFICATION OF ACCEPTANCE

1 message

Microsoft CMT <noreply@msr-cmt.org>
To: Himanshu S Shetty <himanshushettykt03@gmail.com>

3 October 2025 at 01:44

Dear Himanshu S Shetty,

Greetings from the 2025 IEEE Second International Conference on Computing, Semiconductor, Mechatronics, Intelligent Systems and Communications (COSMIC - 2025) organizing committee. We extend our heartfelt appreciation for your submission to our conference, set to take place at Sahyadri College of Engineering and Management, Mangalore.
It brings us great pleasure to inform you that your paper, titled "A Blockchain-Based Framework for Secure Management of Government and Law Enforcement Records", with Submission ID "368" has been successfully accepted for presentation at the 2025 IEEE International Conference on Computing, Semiconductor, Mechatronics, Intelligent Systems and Communications under the "Computing" track.
To maintain the high standards of academic integrity, we kindly request authors to ensure that the plagiarism level of the camera-ready copy remains below 20%. Papers exceeding this limit will regrettably face rejection, even without the option for further modifications. We also encourage authors to incorporate the valuable suggestions and changes provided by our reviewers, as this collaborative effort enhances the overall quality of our conference proceedings.

We will provide further details about the registration and submitting the camera-ready version of your paper, in the next email.

We eagerly anticipate your participation and look forward to welcoming you to the Sahyadri Campus in Mangalore.

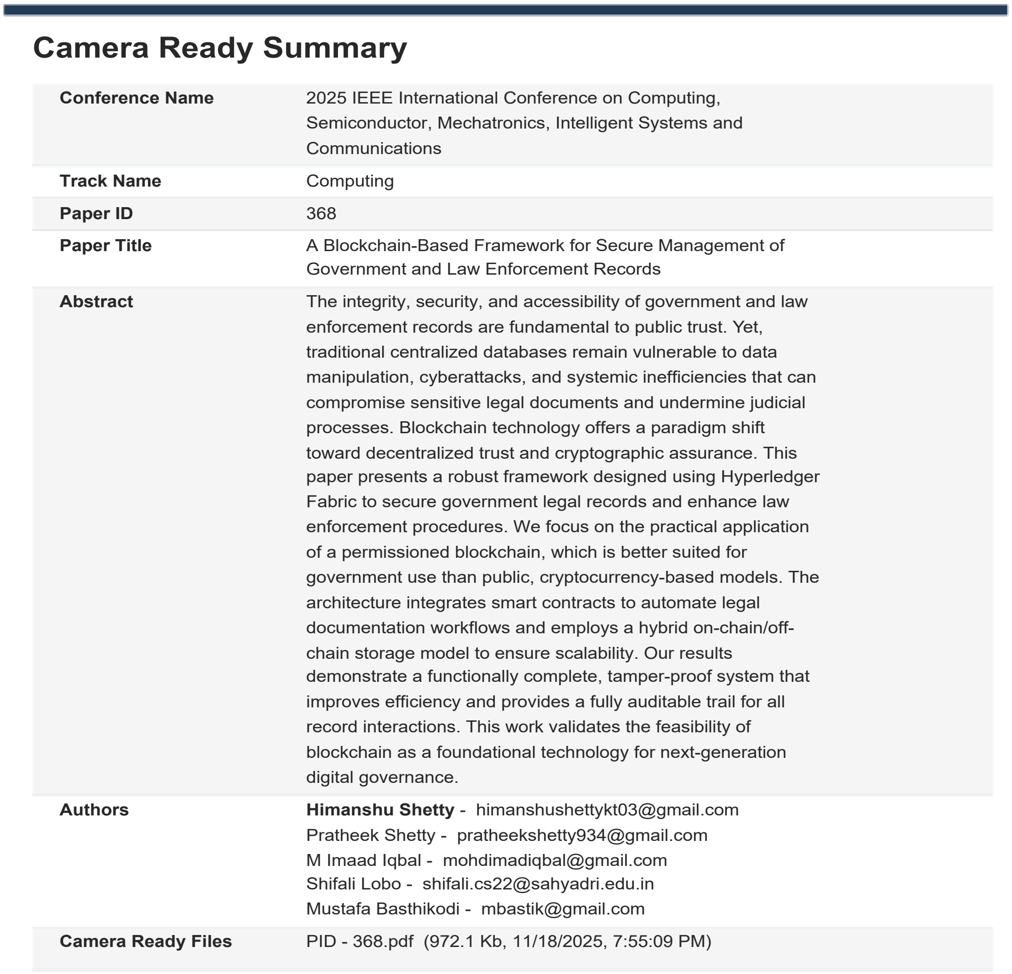
Best Regards,
Organizing Team, IEEE COSMIC-2025
https://cosmic.sahyadri.edu.in

Please do not reply to this email as it was generated from an email account that is not monitored.

To stop receiving conference emails, you can check the 'Do not send me conference email' box from your User Profile.

Microsoft respects your privacy. To learn more, please read our Privacy Statement.

Microsoft Corporation
One Microsoft Way
Redmond, WA 98052



Camera Ready Summary

Conference Name	2025 IEEE International Conference on Computing, Semiconductor, Mechatronics, Intelligent Systems and Communications
Track Name	Computing
Paper ID	368
Paper Title	A Blockchain-Based Framework for Secure Management of Government and Law Enforcement Records
Abstract	The integrity, security, and accessibility of government and law enforcement records are fundamental to public trust. Yet, traditional centralized databases remain vulnerable to data manipulation, cyberattacks, and systemic inefficiencies that can compromise sensitive legal documents and undermine judicial processes. Blockchain technology offers a paradigm shift toward decentralized trust and cryptographic assurance. This paper presents a robust framework designed using Hyperledger Fabric to secure government legal records and enhance law enforcement procedures. We focus on the practical application of a permissioned blockchain, which is better suited for government use than public, cryptocurrency-based models. The architecture integrates smart contracts to automate legal documentation workflows and employs a hybrid on-chain/off-chain storage model to ensure scalability. Our results demonstrate a functionally complete, tamper-proof system that improves efficiency and provides a fully auditable trail for all record interactions. This work validates the feasibility of blockchain as a foundational technology for next-generation digital governance.
Authors	**Himanshu Shetty** - himanshushettykt03@gmail.com Pratheek Shetty - pratheekshetty934@gmail.com M Imaad Iqbal - mohdimadiqbal@gmail.com Shifali Lobo - shifali.cs22@sahyadri.edu.in Mustafa Basthikodi - mbastik@gmail.com
Camera Ready Files	PID - 368.pdf (972.1 Kb, 11/18/2025, 7:55:09 PM)

APPENDIX - C : COPY OF PAPER PUBLISHED

A Blockchain-Based Framework for Secure Management of Government and Law Enforcement Records

Himanshu S Shetty
Dept. of Computer Science and Engineering
Sahyadri College of Engineering & Management
Mangaluru, India
himanshushettykt03@gmail.com

Pratheek G Shetty
Dept. of Computer Science and Engineering
Sahyadri College of Engineering & Management
Mangaluru, India
pratheekshetty934@gmail.com

M Imaad Iqbal
Dept. of Computer Science and Engineering
Sahyadri College of Engineering & Management
Mangaluru, India
mohdimadiqbal@gmail.com

Shifali Florine Lobo
Dept. of Computer Science and Engineering
Sahyadri College of Engineering and Management
Mangaluru, India
shifalilobo9@gmail.com

Mustafa Basthikodi
Dept. of Computer Science and Engineering
Sahyadri College of Engineering and Management
Mangaluru, India
mbasthik@gmail.com

***Abstract*—The integrity, security, and accessibility of government and law enforcement records are fundamental to public trust. Yet, traditional centralized databases remain vulnerable to data manipulation, cyberattacks, and systemic inefficiencies that can compromise sensitive legal documents and undermine judicial processes. Blockchain technology offers a paradigm shift toward decentralized trust and cryptographic assurance. This paper presents a robust framework designed using Hyperledger Fabric to secure government legal records and enhance law enforcement procedures. We focus on the practical application of a permissioned blockchain, which is better suited for government use than public, cryptocurrency-based models. The architecture integrates smart contracts to automate legal documentation workflows and employs a hybrid on-chain/off-chain storage model to ensure scalability. Our results demonstrate a functionally complete, tamper-proof system that improves efficiency and provides a fully auditable trail for all record interactions. This work validates the feasibility of blockchain as a foundational technology for next-generation digital governance.**

***Index Terms*—Blockchain, Hyperledger, Legal Records, Law Enforcement, Smart Contracts, Data Security, Government Applications, Tamper-Proof Systems, Judicial Data, Auditability.**

I. Introduction

In the digital governance era, one of the biggest challenges before public institutions is how to protect the sanctity of official records while making them accessible. Governments switching over from paper-based archives to digital infrastructure expose themselves to a new class of threats that includes sophisticated data breaches, unauthorized tampering, and systemic failures [5], [8]. In the sectors of law and order, this vulnerability assumes grave dimensions. A compromised chain of evidence or a manipulated judicial record can trigger erosion of due process, obstruction of justice, and catastrophic damage to public confidence in the rule of law. Legacy systems are invariably incapable of combating such modern day threats; the shift to a more resilient and accountable digital infrastructure is not just a technological upgrade but a basic imperative for governmental legitimacy [10].

The solution described in this paper is based on blockchain technology-a decentralized ledger system, secure and transparent by design. Going beyond the cryptocurrency roots of blockchain [2], we utilize an enterprise-grade, permissioned framework, Hyperledger Fabric, which is designed to meet specific privacy and governance needs for institutions [3]. The immutable ledger and decentralized consensus at the heart of blockchain's core architectural principles provide a potent guard against data manipulation. Additionally, smart contracts integrate seamlessly and can automate and enforce legal and administrative protocols that reduce the possibilities of fraud and human error, while streamlining very complicated workflows [16], [18]. This proposed approach promises to build a new foundation of trust in judicial processes.

Beyond safety, advantages extend to the operational ability of a blockchain-based system by creating a fully auditable, transparent, chronological track of every transaction or data interaction-a feature very useful in both legal proceedings and internal audits [1]. This has already been exemplified by pioneering real-world implementations: Estonia integrated blockchain technology into its e-governance services to secure public records, while Georgia successfully deployed a similar solution in its land registry to reduce fraud [6, 7].

Despite this promise, there is still a gap between conceptual frameworks and holistic scalable solutions designed to meet

979-8-3315-7364-5/25/$31.00 ©2025 IEEE