Blockchain-Based Document Verification System

1. Abstract

This project implements a decentralized document verification system using Solidity smart contracts, Ethereum blockchain (Ganache), and IPFS decentralized storage. The goal is to replace traditional, easily falsified verification processes with a tamper-proof, immutable, and transparent solution.
The system allows exporters/organizations to upload documents, store their cryptographic hash on the blockchain, and upload the actual file to IPFS. Any user can later verify the authenticity of a document by matching its hash with the blockchain record.

---

2. Introduction

Document fraud remains a major challenge across industries such as education, import–export, healthcare, and legal systems.
This project introduces a blockchain-based approach to ensure:

- Authenticity
- Integrity
- Transparency
- Decentralization

By combining **Blockchain** (for hash storage) and **IPFS** (for file storage), the system allows secure document issuance and verification without depending on any centralized authority.

---

3. Technology Stack

Blockchain Layer

- **Solidity (Smart Contract):** Document storage and exporter management
- **Ganache Local Blockchain:** Development and testing
- **Web3.js:** Smart contract interaction
- **MetaMask:** Wallet & transaction signing

Storage Layer

- **IPFS (Kubo Node):** Decentralized file storage
- Access through local node or public gateways

Frontend Layer

- **HTML, CSS, JavaScript**
- **jQuery for UI states**
- **QRCode.js** for generating verification QR codes

---

4. System Architecture

The application consists of three major layers:

1. Blockchain Layer

- Smart contract manages:
    - Exporters
    - Document hashes
    - IPFS CIDs
    - Verification data

2. IPFS Storage Layer

- The actual document is stored on IPFS
- The blockchain stores **only the hash + CID**
- Ensures immutability and decentralization

3. Frontend Layer

- User uploads file → SHA3 hash generated
- File uploaded to IPFS → CID returned
- Hash + CID written to blockchain
- Verification page fetches data using Web3.js

---

5. Functional Modules

5.1 Smart Contract (DocumentVerification.sol)

Roles:

- **Owner (Admin)**
- **Exporters**
- **Public Users** (verification only)

Key Functions:

- `add_Exporter(address, info)`
- `alter_Exporter(address, newInfo)`
- `delete_Exporter(address)`
- `addDocHash(bytes32 hash, string ipfsCID)`
- `findDocHash(bytes32 hash)`
- `deleteHash(bytes32 hash)`
- `count_Exporters()`
- `count_hashes()`
- `getExporterInfo(address)`

Security

- Only exporter who added a document can delete it
- Only owner can manage exporters
- Hashes uniquely identify documents

5.2 Frontend (HTML + JS + Web3.js)

The user interface supports:

MetaMask login
Hash generation using SHA3
Document upload to IPFS (local Kubo node)
Transaction signing and confirmation
Verification result display
QR Code generation for instant verification
Display of recent uploaded documents

The UI dynamically responds to:

- Network changes
- Account changes
- Smart contract events

5.3 Local IPFS Node Integration

- Used **Kubo v0.30.0**

File uploaded using

```
http://127.0.0.1:5001/api/v0/add
```

- Returns a **CID**, which is stored on the blockchain

File accessible on

```
http://127.0.0.1:8080/ipfs/<CID>
```

- or any public gateway.

---

6. Features and Workflow

Add Exporter – Admin manages exporter database
Exporter Uploads Document
• Hash generated locally
• File uploaded to IPFS
• Hash + CID stored on blockchain
Verify Document
• User uploads document again
• Hash is compared with blockchain
Delete Document (only by exporter)
View Recent Transactions
QR-Code Based Verification Link

---

7. Security and Limitations

Security Strengths

- Document hashes are immutable
- No central database → no tampering possible
- Exporters authenticated through blockchain
- SHA3 hashing eliminates duplication
- IPFS ensures decentralized storage

Limitations

- Requires MetaMask for all interactions
- Local IPFS daemon must be running
- No encryption for documents (future scope)
- Verification depends on user's ability to re-upload the document

---

8. Testing Overview

- Manual testing performed with Ganache accounts
- Tested flows include:
    - Adding/Deleting exporters
    - Hash generation
    - Uploading to IPFS
    - Sending transactions

    - Document verification
    - QR code verification

Future improvements:

- Jest testing for contract interactions
- Automated hash comparison tests

---

9. Deployment and Configuration

Local Setup

1. Start Ganache
2. Start local IPFS daemon
3. Deploy contract via Remix
4. Update contract address in JS config
5. Launch frontend via live-server or VSCode extension

Optional Deployment

- AWS S3 static hosting
- GitHub Pages
- IPFS-hosted frontend

---

10. Conclusion

This project demonstrates a **secure, transparent, decentralized document verification system** powered by Blockchain and IPFS.
It ensures tamper-proof storage, verifiable authenticity, and eliminates centralized control.
The solution is highly scalable and can be extended for:

- Universities
- Government Authorities
- Export Agencies
- Healthcare and Legal documentation

It serves as a strong foundation for real-world blockchain adoption.