IoB Project VisaChain: A Theoretical Private Ethereum Blockchain using P2P transfer and Smart Contracts for Embassies

Encadrant : **Maria POTOP-BUTUCARU**

Etudiants : **Kerrian AZIZA, Ihsane BOUBRIK**

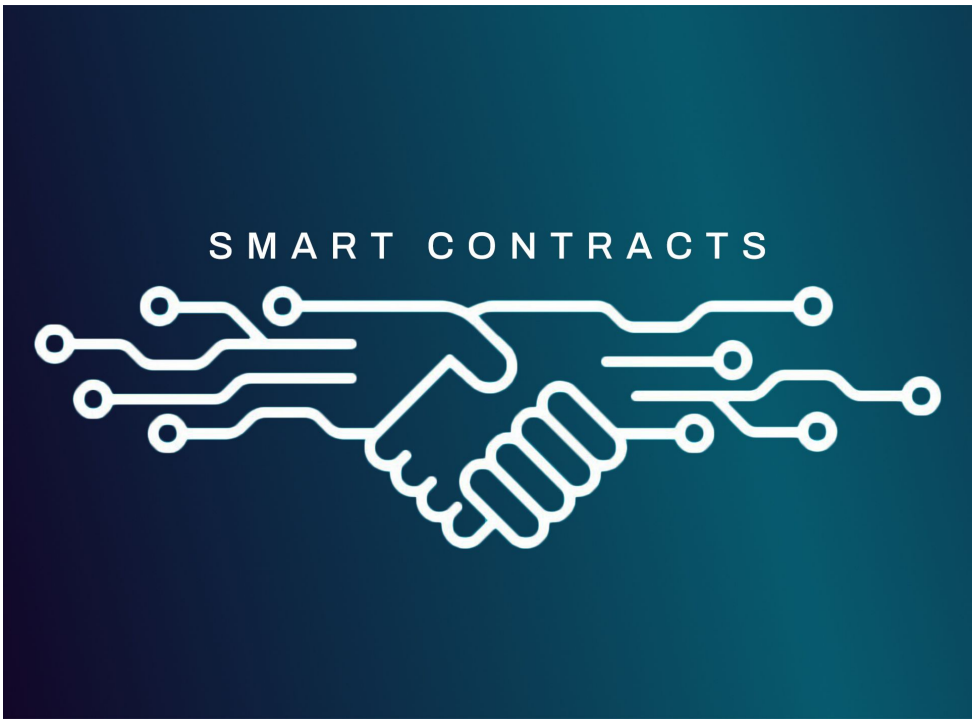

Table des matières

All of the VisaChain code is available through GitHub with a Readme.md explaining how to test it in Remix and IPFS at the following address : https ://github.com/KerrianAZIZA/visachain

1 Introduction

The increasing digitalization of administrative procedures has led to a growing need for secure, transparent, and reliable document management systems. In particular, documents related to visas, identity verification, and administrative authorizations are highly sensitive and often exchanged between multiples institutions such as embassies, consulates and governmental authorities. Traditional centralized information systems present several limitations, including single points of failure, restricted transparency, and difficulties in ensuring traceability and data integrity across organizational boundaries.

Blockchain technology offers an alternative approach by enabling decentralized, tamper-resistant record keeping without relying on a single truster authority. Within this context, this project explores the design and implementation of a decentralized document certification system based on a blockchain and a distributed storage network. The proposed system, named *VisaChain* aims to serve as a conceptual prototype for a consortium blockchain shared between embassies, where the blockchain acts as a certification and audit layer for document exchanges.

Rather than storing documents directly on-chain, which would be inefficient and costly, the system relies on the InterPlanetary File System (IPFS) for decentralized file storage. Only cyptographic references to the documents, in the form of Content Identifiers (CIDs) are stored on the blockchain. This approach ensures document integrity while preserving scalability and reducing on-chain storage costs.

This report is structured as follows. Section 2 presents the context and objectives of the system, with a focus on document sharing between embassies. Section 3 introduces IPFS and explains its role within the architecture. Section 4 details the functional analysis and design choices. Section 5 describes the implementation of the VisaChain smart contract. Section 6 presents the experimental setup and demonstration scenario. Finally, Section 7 discusses the limitations of the prototype and potential improvements.

2 Context and System Objectives

2.1 The document sharing problem between embassies

The management of administrative documents between embassies and consulates is inherently complex due to the sensitivity, sheer volume and the diversity of the documents involved. These documents such as Visa applications, identity verification records and other official authorizations often contain highly confidential and personal information. Any unauthorized access or accidental modification or even data loss could have severe legal and operational consequences. It is therefore critical to ensure the integrity, confidentiality and availability of such documents.

Traditional centralized information systems rely on a single authority to maintain and control access to the database. While functional for limited internal use, these systems systems suffer from multiple key weaknesses when applied to multi-stakeholder environments :

- **Single Point of Failure** : A central server can be compromised by cyberattacks, hardware failures or insider threats. If the server is unavailable, all document access is disrupted.
- **Limited Transparency :** Centralized systems typically do not provide verifiable records of document modifications, making it difficult to audit actions across multiple stakeholders.
- **Access Control Complexity :** Different stakeholders require different levels of access (read-only, edit, validate) and enforcing these permissions consistently is challenging in traditional systems.
- **Cross-Border Coordination :** Embassies operating in different countries must comply with local regulations while collaborating internationally, increasing operational complexity.

Furthermore, the traditional approach often results in duplicated copies of documents, increasing the risk of inconsistencies and version conflicts. Maintaining a consistent record that all stakeholders can trust is therefore a significant challenge.

In summary, the current centralized models are inadequate for scenarios where multiple, semi-trusted parties must collaboratively manage sensitive documents with verifiable history and strict access control. This motivates the exploration of a decentralized, blockchain-based solution.

2.2 Theoretical vision : a consortium blockchain for embassies

To address these challenges, the VisaChain project proposes a consortium blockchain model in which multiple embassies operate as semi-trusted nodes within a shared network.

The blockchain functions as an immutable distributed ledger that records metadata, cryptographic hashes and action logs associated with each document. This structure ensures transparency, traceability and integrity without relying on a single trusted authority.

In this architecture, each document is represented on-chain by a cryptographic reference (CID) pointing to its content stored in the InterPlanetary File System (IPFS). The blockchain does not store the document content itself, which preserves scalability and confidentiality. Instead, it records key information about the document lifecycle including :

- Creation and author information
- Version history
- Assigned roles and permissions
- Validation and certification events by authorized personnel.

Each actor in the system is assigned a role (owner, reader, editor, validator) with explicit permissions for each document. For instance, an embassy officer may be authorized to edit or validate a document, while another embassy may have read-only access. This approach ensure that all actions are cryptographically verifiable and can't be repudiated.

By combining the blockchain with IPFS, the system guarantees **integrity**, **availability**, **auditability** and **confidentiality**. This design is particularly suited to a consortium of embassies where multiple semi-trusted parties must collaborate while maintaning independent control over their nodes. The blockchain acts as neutral certification layer, ensuring that all actions taken by any embassy are transparent and verifiable without revealing sensitive information to unauthorized parties.

In essence, the theoretical goal of VisaChain is to provide a secure, verifiable, and decentralized document sharing platform, capable of meeting the strict operational and regulatory requirements of international administrative procedures. The combination of blockchain and IPFS offers a practical solution to the challenges of multi-stakeholder document management, balancing security, transparency, and efficiency.

3 Decentralized Storage and Data Integrity with IPFS

3.1 A P2P Network for the document storage

While blockchains provide immutability and transparency and trustless verification, sorting large files directly on-chain is impractical for servral reasons. First, blockchain storage is highly expensive which means that every byte stored on-chain incurs gas costs, making large documents such as visa applications or administrative folders financially prohibitive. Second, on-chain storiage has size limitations, as blocks have a maximum capacity and large data inflates the blockchain size which negatively impacts node synchronization and network performance.

Furthermore and most importantly, sensitive documents stored on a public blockchain would be visibile to all network participants violating confidentialy requirements. Enven in permissioned or consortium blockchains, storing raw document data on-chain is unnecessary and inefficient as the blockchain's primary function is to maintain verifiable references and audit logs not to serve as a file server.

For these previous reasons, VisaChain separates document storage from document verification. Only a cryptographic hash (CID) of the document is stored on-chain. The actual document content is stored off-chain in a distributed file system IPFS.

3.2 InterPlanetary File System (IPFS)

IPFS is a peer-to-peer distributed file system designed to address the limitations of centralized file storage. It uses content-addressing rather than location-based addressing. Each file is identified by the CID, a cryptographic hash of the cile content. This approach ensures that any modification of the file changes its CID, making it tamper-evident.

Key characteristics of IPFS include :

- Decentralization : files are stored across multiple nodes, reducing reliance on a central server and enhancing fault tolerance.
- Immutability : Once a file is added to IPFS, its CID permanently references the specific content version. This property guarantees integrity and traceability of documents.
- Efficient retrieval : IPFS located content based on its CID, allowing nodes to fetch files from the nearest or fatest available peer, optimizing bandwith.
- IPFS can maintain multiple versions of a file, which aligns naturally with VisaChain's need for document version tracking.

By leveraging IPFS, VisaChain ensures that documents remain securely stored, verifiable and resistant to tampering while avoiding the high cost of on-chain storage.

3.3 The Ethereum-IPFS integration

In VisaChain, IPFS is integrated with Ethereum to combine the centralized storage capabilities of IPFS with the ummutability and auditability of the blockchain. The workflow can be summarized as follows :

- Document Upload : The user uploads a document to their local IPFS node. IPFS returns a unique CID representing the document.
- On-Chain Registration : The CID, along with document metadata is stored on the Ethereum blockchain using a smart contract.
- Access Control : Smart contracts enforme permissions for reading, editing and validation documents based on the assigned roles. Only authorized actors can access the document's CID and fetch the file from IPFS.
- Verification : Any actor retrieving the document from IPFS can verify its integrity by comparing the file's hash with the CID stored on-chain. Any discrepancy indicates tampering.

4 Functional Analysis and System Design

4.1 Actors and roles

The VisaChain system involves multiple actors, each with distinct responsibilities and privileges to ensure secure and controlled document management. The primary roles are defined as follows :

- Document Owner : The user who created the document on the blockchain. The owner has full control over the document, including the ability to assign roles, update content and revoke the document if necessary. The owner ensures the document's lifecycle is properly managed.
- Editors : Users granted permission to modify or update the document. Editors can create new versions of a document, including updated files or descriptive notes, while the blockchain maintains a verifiable version history. Editors do not have the authority to validate or certify documents unless explicitly assigned as validators.
- Readers : Users authorized only to view the document. Readers can access document metadata and fetch the file from IPFS. They can't however alter its content or validate it. This role ensures that sensitive documents are accessible to authorized parties without risking unauthorized modifications.
- Validators : Users responsible for certifying the authenticity or approval of a document. Validators can "sign" the document marking it as officially verified. The blockchain records these signatures in an immutable log, provinding traceability and accountability. Validators may also have read or edit permissions as determined by the document owner.

4.2 Document lifecycle

The lifecycle of a document in VisaChain is designed to reflect typical administrative workflows while leveraging blockchain's immutability and IPFS storage. The main stages. These main stages are :

- The creation : A user(owner) creates a document, uploads the file to IPFS and store the CID on-chain along with metadata, including a descriptive note. The document initially enters the *Draft* state.
- Versioning : Editors authorized by the owner can update the document. Each update generates a new version, preserving the previous versions on-chain as part of the history. Yhis mechanism prevents data loss and ensures accountability for all modifications
- The Validation/Certification : Validators can sign the document to certify its authenticity. Once signed, the document's status may transition to *Active* to indicate

its verified state. Each signature is permanently recorded on the blockchain establishing an audit trail.

- Revocation : The document owner retains the ability to revoke a document at any time. Revocation changes the document's status to *Revoked* and prevents further modifications or validation, ensuring that obsolete or invalid documents are not misused.

This lifecycle ensures that documents are securely tracked, auditable and managed, reflecting real-world administrative requirements while maintaining transparency and immutability.

4.3 Overall architecture

The architecture of VisaChain integrates three primary components :

- Smart Contract Layer (Ethereum) : Enforces document lifecycle rules, permissions, versioning and validator signatures. This layer provides immutability and trustless verification of all actions.
- Decentralized Storage (IPFS) : Stores the actual document content. The content's integrity is verifiable via the CID stored on-chain.
- Theoretical User Interface : In concept, a user interface which provides a meachnism for embassies or users to upload documents, assign roles, view summaries and validate documents. In practice, this is not yet programmed and not included in the project. Such interface with both IPFS nodes and smart contract would ensure smoother workflows.

5 The Smart Contract Implementation

5.1 Overview

The VisaChain smart contract implements the core business logic of the document management system, including document lifecyle management, role-based access controle, versioning and validation tracking. The contract is on an Ethereum-compatible blockchain, where it leverages the blockchain's immutability and transparency to guarantee traceability and trusthworthiness of all actions. It is written in Solidity.

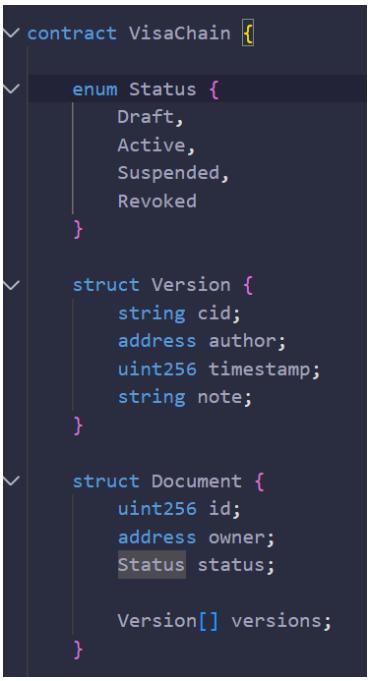

Figure 1 : The declaration part of VisaChain.sol

As seen in Fig.1 each document has 4 status, a version with a CID, an auhtor, the timestamp and a note to give more metadata on the file. The document is given an ID, the docID, this ID is only for the blockchain part and this ID is used to verify and retrieve the CID from the blockchain.

5.2 Document Managements Functions

The VisaChain smart contract provides several key functions for managing documents.

The createDocument function in Fig.2 allows the owner to create a new document by uploading the resulting CID of the IPFS upload with a descriptive note. The document enters the *draft* state. The smart contract assign the creating user as the document owner and emits a DocumentCreated event.

The updateDocument function in Fig.3 allows authorized editors to update the document, generating a new version with a new CID and an optional note. Each update

```
function createDocument(string memory cid, string memory note) external {    infinite gas
    documentCount++;

    Document storage d = documents[documentCount];
    d.id = documentCount;
    d.owner = msg.sender;
    d.status = Status.Draft;

    d.versions.push(
        Version({
            cid: cid,
            author: msg.sender,
            timestamp: block.timestamp,
            note: note
        })
    );

    emit DocumentCreated(documentCount, msg.sender, cid);
}
```

Figure 2 : CreateDocument function of VisaChain.sol

```
function updateDocument(    infinite gas
    uint256 docId,
    string memory newCid,
    string memory note
) external canEdit(docId) {
    Document storage d = documents[docId];
    require(d.status != Status.Revoked, "Document revoked");

    d.versions.push(
        Version({
            cid: newCid,
            author: msg.sender,
            timestamp: block.timestamp,
            note: note
        })
    );

    emit DocumentUpdated(docId, msg.sender, newCid, note);
}
```

Figure 3 : UpdateDoc function of VisaChain.sol

preserves previous versions in the array, maintaining the version history. This function also emits a DocumentUpdated event for transparency and tracking

```
function revokeDocument(uint256 docId) external onlyOwner(docId) {    28319 gas
    documents[docId].status = Status.Revoked;
    emit DocumentRevoked(docId);
}
```

Figure 4 : RevokeDoc function of VisaChain.sol

The revokeDocument allows the owner to revoke a document at any time, setting its status to *Revoked*. Once revoked, the document can no longer be modifier or validated, preventing misues of obsoltete or invalid records. The revocation event is loggend on-chain.

The transferOwnership function allows owners to transfer the ownership of a document to another address, providing flexibility for administrative changes. This function updates

```
function transferOwnership(uint256 docId, address newOwner)    infinite gas
    external
    onlyOwner(docId)
{
    require(newOwner != address(0), "Invalid address");
    address oldOwner = documents[docId].owner;
    documents[docId].owner = newOwner;

    emit OwnershipTransferred(docId, oldOwner, newOwner);
}
```

Figure 5 : transferOwnership function of VisaChain.sol

the owner field and emits an OwernshipTransferred event.

```
function getDocumentSummary(uint256 docId)    infinite gas
    external
    view
    canRead(docId)
    returns (
        uint256 id,
        address owner,
        Status status,
        string memory latestNote
    )
{
    Document storage d = documents[docId];
    Version storage v = d.versions[d.versions.length - 1];

    return (d.id, d.owner, d.status, v.note);
}
```

Figure 6 : getDocSummary function of VisaChain.sol

The getDocumentSummary function allows a reader user to retrieve a document's latest version information, including CID, descriptive note, author and timestamp.

5.3 Permission management

VisaChain enforces role-based permissions using a combination of mapping and address arrays :

- Editors : Can create new versions of a document.
- Readers : can view the document's metadata and fetch its content from IPFS.
- Validators : can sign the document to certify its authenticity.

The smart contract includes function to assign and revoke these roles dynamically. Only the document owner has the authority to modify roles, ensuring consistent and secure management. This approach guarantees that only authorized actions are executed on the blockchain while maintaining a complete audit trail

```
function setEditor(uint256 docId, address user, bool allowed)    78521 gas
    external
    onlyOwner(docId)
{
    if (allowed && !editors[docId][user]) {
        editorList[docId].push(user);
    }
    editors[docId][user] = allowed;
}

function setValidator(uint256 docId, address user, bool allowed)    78544 gas
    external
    onlyOwner(docId)
{
    if (allowed && !validators[docId][user]) {
        validatorList[docId].push(user);
    }
    validators[docId][user] = allowed;
}

function setReader(uint256 docId, address user, bool allowed)    78566 gas
    external
    onlyOwner(docId)
{
    if (allowed && !readers[docId][user]) {
        readerList[docId].push(user);
    }
    readers[docId][user] = allowed;
}
```

Figure 7 : Permission functions of VisaChain.sol

5.4 Validation and signature logic

The validation mechanism is implemented as follows :

- Each validator can sign a document once, marking it as certified.
- The HasSigned mapping tracks whether a validator has approved a document.
- A document may transition to the *Active* state after the first signature, reflecting its verified status.
- Owners can query which validators have signed and which have not, enabling full transparency for certification processes.

The function signDocument in Fig.8 uses the docID to sign the document only if the user has the validator role, otherwise the signature will failed.

```
function signDocument(uint256 docId) external {    55532 gas
    require(validators[docId][msg.sender], "Not validator");
    require(!hasSigned[docId][msg.sender], "Already signed");

    hasSigned[docId][msg.sender] = true;

    // Active dès la première signature (modifiable)
    documents[docId].status = Status.Active;

    emit DocumentSigned(docId, msg.sender);
}
```

Figure 8 : Sign function of VisaChain.sol

```
    function getValidatorSignatures(uint256 docId)    infinite gas
        external
        view
        onlyOwner(docId)
        returns (address[] memory validators_, bool[] memory signed_)
    {
        address[] memory vList = validatorList[docId];
        bool[] memory sList = new bool[](vList.length);

        for (uint256 i = 0; i < vList.length; i++) {
            sList[i] = hasSigned[docId][vList[i]];
        }

        return (vList, sList);
    }
}
```

Figure 9 : GetValidatorsSignature function of VisaChain.sol

The function getValidatorsSignatures in Fig.9 uses also the docID and if the user is the owner then it will return the list of all validators and if they have signed or not the document corresponding to the docID.

5.5 Read and query functions

VisaChain provides several read-only functions to facilitate document access and audit :

The first function is getLatestVersion in Fig.10 which returns the CID, note and author and timestamp of the most recent version of the document using the docID.

The second function getVersionCount in Fig.11 returns the number of versions of a document based on the docID.

The functions getValidators and getReaders in Fig.12 return arrays of assigned users for each roles.

```
function getLatestVersion(uint256 docId)    infinite gas
    external
    view
    canRead(docId)
    returns (
        string memory cid,
        string memory note,
        address author,
        uint256 timestamp
    )
{
    Version storage v =
        documents[docId].versions[
            documents[docId].versions.length - 1
        ];

    return (v.cid, v.note, v.author, v.timestamp);
}
```

Figure 10 : GetLatestVersion function of VisaChain.sol

```
function getVersionCount(uint256 docId) external view returns (uint256) {    infinite gas
    return documents[docId].versions.length;
}
```

Figure 11 : GetVersionCount function of VisaChain.sol

6 Experimental Setup and Demonstration

6.1 Experimental environment

To evaluate the proposed VisaChain system, a local experimental environment was deployed, combining an Ethereum-compatible blockchain through Remix Desktop and multiples IPFS nodes. The objective was to simulate a realistic multi-actors scenario similar to interactions between embassies or adminstratuve entities.

The blockchain environment was based on a Ethereum Testnet through Remix Desktop, allowing fast deployment withou transactions costs. Each participants interacted with the smart contract using a distinct Ethereum address, representing independ institutional actors.

For decentralized storage, the IPFS was used. Three separate IPFS nodes were configured to simulate three different users (User A, User B and User C). Each node operated with its own repository path, API port and gateway prt, ensuring full isolation between participants.

```
function getValidators(uint256 docId)    infinite gas
    external
    view
    onlyOwner(docId)
    returns (address[] memory)
{
    return validatorList[docId];
}

function getReaders(uint256 docId)    infinite gas
    external
    view
    onlyOwner(docId)
    returns (address[] memory)
{
    return readerList[docId];
}
```

Figure 12 : GetValidators and GetReaders functions of VisaChain.sol

6.2 Demonstration scenario

User A created a test document locally and uploaded it to their IPFS node. Upon upload, IPFS generated a CID derived from the hash of the file.

This CID is unique as said previously and is only for now in User A's possesion.

```
kerrian@LaptopKerrian:~$ export IPFS_PATH=~/ipfs_userA/.ipfs
echo "Ceci est un document secret" > testfile.txt
ipfs add testfile.txt
added QmVrys5X5PcVsiLeYXWhTm1HkdEgTJEY81ujz6DP74Bm5N testfile.txt
kerrian@LaptopKerrian:~$
```

Figure 13 : IPFS upload from User A

Then in a second time, the user A will deploy the VisaChain smart contract and create a new document by submitting the CID and a descriptive note. The document will be stored on-chain in the *draft* state.

User A will then assigned the role of reader to user B and editor/validator to user C.

As we can see in document count in Fig.15, for now there is only one document on the blockchain so the docID of the test is 1.

Once the roles are assigned User B can retrieve the latest version of the document in Fig.16 and in a second time with the CID retrieve the file in IPFS in Fig.17

In the same time, user C can sign the document and make modifications through the blockchain. C will use the signDocument function and user A will be able to see the list of validators and their signing status as seen in Fig.18 where the address of User C is marked with the boolean true.

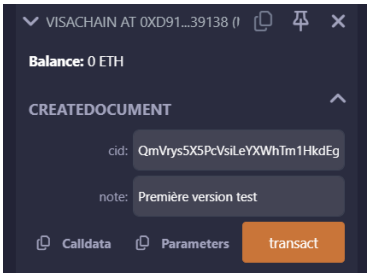

Figure 14 : Deployment from User A

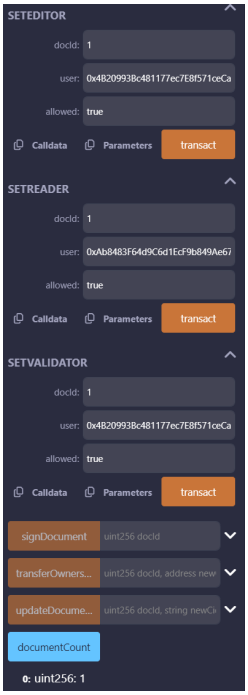

Figure 15 : Role assignment from User A

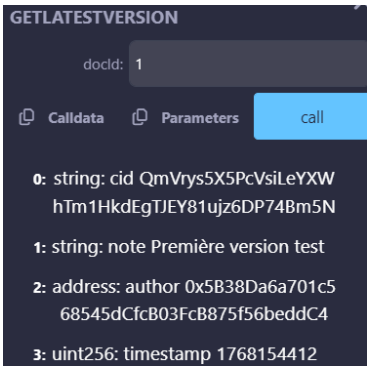

Figure 16 : CID Retrievement from User B

```
kerrian@LaptopKerrian:~$ export IPFS_PATH=~/ipfs_userB/.ipfs
ipfs cat QmVrys5X5PcVsiLeYXWhTm1HkdEgTJEY81ujz6DP74Bm5N
Ceci est un document secret
kerrian@LaptopKerrian:~$
```

Figure 17 : File Retrievement in IPFS from User B

Figure 18 : GetValidatorsSignature from the POV of User A

This scenario demonstrates secure collaboration between multiple independent actors while preserving integrity, traceability, and access control.

6.3 Results and observations

The experiment validated several key aspects of the system :

- Documents stored on IPFS could be reliably retrieved using on-chain CIDs.
- Unauthorized users were prevented from reading, editing or validation documents.
- Document versioning preserved a complete and immutable history/
- Validators signatures were transparently recorded and verifiable on-chain.
- The system operated without relying on a centralized authority or storage provider.

These results confirm that VisaChain effectively enforces access control, integrity, and certification in a decentralized environment.

7 Limitations and Future Work

7.1 Limitations of the prototype

One of the main limitations of the current system is the absence of content confidentiality at the storage level. Documents stored on IPFS are publicly addressable via their CID. While access control is enforced on-chain, IPFS itself does not prevent unauthorized users from retrieving a file if they obtain the CID. This means that confidentiality relies primarily on off-chain trust and secrecy of the CID.

Another limitation concerns signature logic. In the current implementation, a document becomes active as soon as a single validator signs it. While this simplifies the workflow, real-world administrative processes may require multiple signatures or a predefined quorum of validators before a document is considered valid.

Scalability is also a consideration. Each document stores an array of versions on-chain. While this ensures full traceability, it increases storage costs and may become inefficient for documents with a large number of revisions. This trade-off is acceptable for a prototype but would require optimization in a production environment.

Finally, the system currently lacks a user-friendly interface. All interactions are performed via command-line tools or scripts. This limits usability for non-technical users, such as administrative staff in embassies.

7.2 Security Considerations

From a security perspective, the smart contract enforces strict access control through role-based permissions. However, some improvements could further strengthen the system :

- There is no cryptographic binding between the signer and the document content beyond the CID reference. While the CID ensures content integrity, additional metadata signing could strengthen non-repudiation.
- The contract does not prevent role abuse by the owner. In a real consortium, governance rules or multi-signature ownership may be required.
- Revocation does not invalidate previously downloaded files. Once a document is retrieved from IPFS, it cannot be technically revoked from the network.

These aspects reflect inherent trade-offs in decentralized systems and must be addressed at both technical and organizational levels.

7.3 Future Improvements

Several enhancements could significantly improve VisaChain :

- Encrypted Storage :Documents could be encrypted before being uploaded to IPFS. Decryption keys could then be shared securely only with authorized readers, ensuring confidentiality even if the CID is leaked.

- Multi-Signature Validation : A threshold-based validation mechanism could require approval from multiple validators before a document becomes active. This would better reflect diplomatic or inter-institutional procedures.
- Front-End Application : A web-based interface would greatly improve usability, allowing users to upload documents, assign roles, and validate files without direct interaction with the blockchain or IPFS CLI.
- Interoperability The system could be extended to support multiple blockchain networks or cross-chain verification for broader international adoption.

7.4 Applicability to Embassy Use Cases

Despite its limitations, VisaChain demonstrates strong potential for embassy and diplomatic applications. The system provides :

- A neutral and tamper-proof certification mechanism
- Decentralized document storage
- Transparent validation and auditability
- Reduced reliance on centralized authorities

These properties align well with inter-embassy collaboration, where trust must be distributed and verifiable without a single controlling entity.

8 Conclusion

This project presented VisaChain, a decentralized system designed to securely manage, share, and certify sensitive documents using blockchain and IPFS technologies. The proposed solution addresses a realistic administrative use case, namely the secure exchange of documents between embassies or institutional entities, where trust, traceability, and integrity are critical requirements.

By leveraging Ethereum smart contracts, VisaChain ensures immutable record-keeping, role-based access control, document versioning, and transparent validation through on-chain signatures. At the same time, IPFS provides an efficient and decentralized storage layer, allowing large files to be stored off-chain while preserving integrity through content-addressed identifiers (CIDs). The combination of these technologies enables a clear separation between data storage and trust enforcement.

The experimental evaluation demonstrated that the system successfully enforces permissions, maintains a verifiable document lifecycle, and supports multi-actor collaboration without relying on a centralized authority. Validators' signatures and document updates are permanently recorded on-chain, providing a reliable audit trail suitable for institutional environments.

While the current implementation remains a prototype, it highlights both the strengths and limitations of decentralized architectures. Identified improvements, such as encrypted storage, multi-signature validation, and consortium governance models, open promising directions for future work.

In conclusion, VisaChain demonstrates that blockchain-based document certification, combined with decentralized storage, is a viable and effective approach for inter-institutional document management. This project illustrates how blockchain technologies can move beyond financial applications to support secure, transparent, and collaborative administrative systems.