COMP4651 Project Report
IPFS Library: Implementation of a Decentralized Data Management Platform Using Blockchain Technology

Runsheng Liu
SID: 20944989
rliuar@connect.ust.hk

Ruiliang Gong
SID: 20942228
rgongac@connect.ust.hk

Junchi Lu
SID: 20945048
jlubp@connect.ust.hk

Abstract

This project explores the design and implementation of a decentralized storage system leveraging InterPlanetary File System (IPFS), OrbitDB, and blockchain technology, named IPFS Library. While IPFS offers robust decentralized file storage, its usability is limited by the need to know file-specific Content Identifiers (CIDs) for retrieval. To address this, we integrated OrbitDB to store metadata such as file titles and authors, enabling users to search and retrieve files intuitively. Blockchain technology enhances transparency by recording user actions, such as file uploads and deletions, while a virtual currency mechanism based on Ethereum incentivizes responsible behavior and optimizes resource use. Our system demonstrates an accessible, accountable, and decentralized solution for file storage, advancing the principles of Web3. The code is available in our GitHub repository, where you can also find a demonstration video showcasing the project.

Keywords: *Decentralized File Storage, Blockchain, Smart Contract*

1 Introduction

Decentralized technologies are revolutionizing data storage and management by addressing the inherent limitations of centralized systems. Traditional solutions, dominated by corporations like Google and Amazon, face criticism for their vulnerability to data breaches, lack of transparency, and concentration of control. As digital ecosystems continue to expand, there is a pressing need for more equitable and secure systems to store and access data, adhering to the principles of Web3 [Wik24].

Decentralized storage offers a compelling alternative by distributing data across a network of nodes, eliminating single points of control and failure. Among existing solutions, the InterPlanetary File System (IPFS) stands out for its innovative content-addressing approach, enabling efficient and decentralized file sharing. However, IPFS suffers from a significant usability challenge: users must know the exact content identifier (CID) to retrieve files, making large-scale adoption cumbersome.

To bridge this gap, our project introduces IPFS Library, a system designed to enhance decentralized storage with intuitive file discovery and governance mechanisms. By integrating metadata-driven search capabilities and incorporating a transparent usage framework, we aim to make decentralized storage not only more accessible but also fair and sustainable.

This paper discusses our approach to addressing the usability challenges of decentralized storage while ensuring a transparent and incentivized ecosystem. The following sections provide an overview of related work, detailing the advancements in decentralized storage, metadata management, and blockchain-based governance, before presenting our methodology and implementation.

2 Related Work

Decentralized file storage has been explored as a solution to the limitations of centralized systems. The InterPlanetary File System (IPFS) [Ben14] is a widely used protocol known for its content-addressing model, which ensures efficient and decentralized file sharing. However, IPFS requires users to know the exact Content Identifier (CID) for retrieval, making it less intuitive for broader adoption.

To address the usability challenges of decentralized storage, tools like OrbitDB [Con24] have been developed. OrbitDB extends IPFS by enabling metadata-driven queries, allowing users to search files by attributes such as titles or authors. Despite its advantages, OrbitDB adds complexity to decentralized storage systems, especially in maintaining consistency across distributed nodes.

Blockchain technology further complements decentralized storage by providing transparency and accountability. Solutions such as Storj and Sia [Lab24] integrate blockchain for governance and resource allocation, but they do not fully resolve the challenges of file discovery and metadata management.

Our project builds on these efforts by integrating IPFS and OrbitDB with a blockchain-based governance model. This approach not only enhances file searchability through metadata but also ensures transparent and equitable system management, addressing key limitations of existing solutions.

3 Methodology

3.1 System Design

The proposed decentralized storage system leverages the InterPlanetary File System (IPFS), OrbitDB, and blockchain technology to achieve a seamless and transparent framework for data storage and retrieval. This section details the architectural design of the system, focusing on its components, interactions, and functionalities. The framework, as illustrated in the accompanying diagram Fig. 1, is divided into two primary functionalities: File Storage and File Search.

3.1.1 System Modules

IPFS Network. The IPFS Network provides a decentralized mechanism for file storage and retrieval. In our project, both the CID and the actual file are stored on the IPFS network. Files are uploaded to the IPFS network, which generates a unique Content Identifier (CID) for each file. The CID is crucial for retrieving the file and ensuring its content integrity.

OrbitDB. The OrbitDB acts as the metadata repository, allowing users to store and query information about uploaded files, such as authors, titles, and associated CIDs. OrbitDB is a peer-to-peer database built on top of IPFS, ensuring the metadata is stored in a decentralized and immutable manner.

Blockchain. Our designed blockchain regards every operation on files as transactions, and it ensures transparency and accountability by logging user behaviors, including file uploads and deletions. Blockchain also incorporates smart contracts to facilitate and enforce resource management, using a virtual currency mechanism to regulate file uploads.

3.1.2 Workflow

The system operates across two primary workflows: File Storage and File Search, with distinct interactions among clients, modules, and contracts.

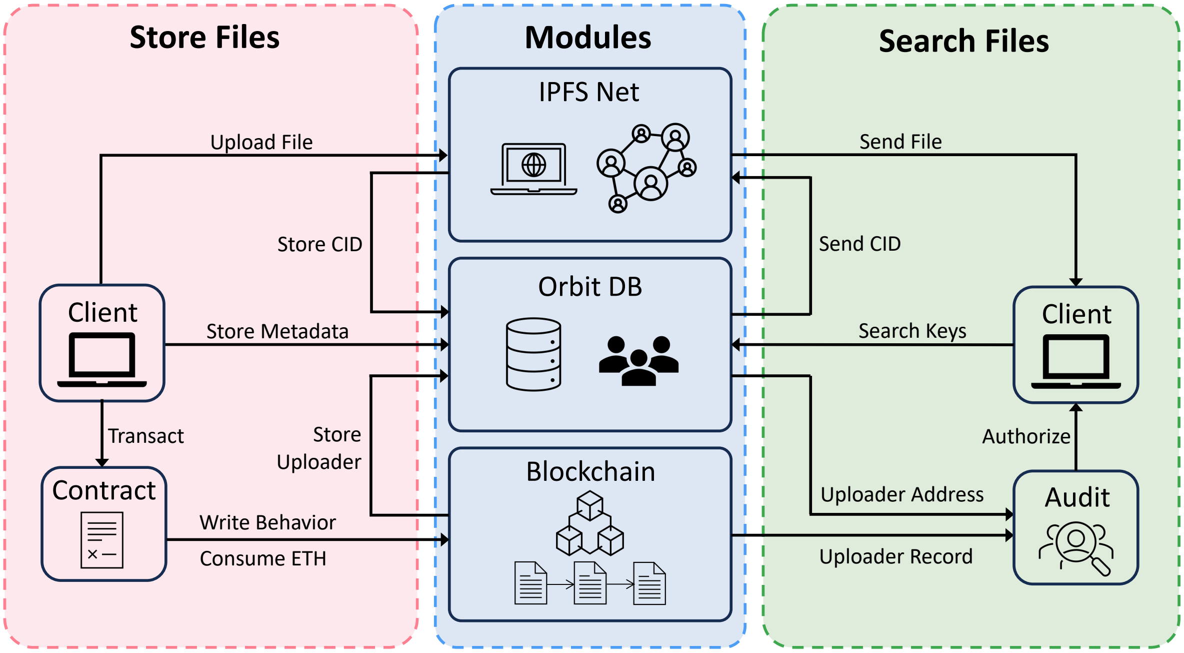

Figure 1: The workflow of IPFS Library. It shows how the two main processes, storing files and searching files, interact with the core modules of IPFS Library.

File Storage. When users upload a file, the system distributes its management across the three core modules. The file itself is stored on the IPFS network, and a CID is generated. The metadata associated with the file, such as the title and author, is indexed and stored in OrbitDB, enabling efficient file lookup. Simultaneously, the blockchain records the upload event, including the user's address and transaction details, ensuring transparency. To promote responsible usage, users are required to pay a small amount of ETH through a smart contract to complete the upload process, discouraging storage abuse and ensuring fair usage.

File Search. The search functionality is designed to prioritize user accessibility. Users can query the system using metadata, such as file titles or authors, instead of having to know the file's CID. OrbitDB retrieves the relevant CID based on the query, and the file is fetched directly from the IPFS network. The system also employs blockchain records to authorize user actions, such as deletions, ensuring that only the original uploader can remove their files. This guarantees secure and transparent data management while preventing unauthorized deletions.

4 Implementation Details

4.1 Tools and Technology Utilized for Development

As a comprehensive decentralized platform integrating interactive user interface, IPFS technologies, database reading and writing, Ethereum [Woo] blockchain technologies, the IPFS Library involves multiple tools for implementation.

Front-end and Back-end Development: React.js and Node.js
We mainly used React.js for designing the user interface, and Node.js serves as the primary runtime environment for server-side operations. It is responsible for multiple functions such as handling IPFS and OrbitDB interactions, managing blockchain communication, processing file operations and so on.

Data Storage: IPFS and OrbitDB

IPFS is a storage system which creates a peer-to-peer network for file distribution resembling a decentralized database system. It was implemented in Kubo (as seen in the `kubo/` directory) for convenient distribution and storage between different nodes. OrbitDB is a decentralized database system managed through the `database/orbitdb.mjs` module. It provides services like traditional database systems, but it stores data in multiple nodes. Up to now, it is only associated with relatively basic operations including query, addition and deletion.

Blockchain: Ethereum Smart Contract
The contract is written by Solidity [1], which is Ethereum's native smart contract programming language.

Truffle and Hardhat[2] frameworks are applied for compiling the contracts. Besides, they are also responsible for deploying contracts to the blockchain, just like lauching applications to the Internet. Moreover, since the contracts distributed on the blockchain are immutable, updating such contracts are all about allocating a new contract, and migrating all the information to the replicate. Truffle and Hardhat frameworks can also manage such migration process.

Web3.js [webnd] is a JavaScript library which functions as an interface between the front-end and the contracts. Its works include connecting to blockchain, reading and writing smart contracts, creating and sending contract to blockchain, listening to events on blockchains for real-time update of the related UI components and so on.

The Foundryup [frnd] package is installed and Anvil is used within it to simulate a local blockchain environment by creating 20 virtual clients, since it is necessary to set up a mock blockchain environment for test and development without incurring actual fees during operations. In the browser, where we can inspect our IPFS Library website, we use MetaMask [3] to efficiently interact with Anvil.

4.2 Code Structure

Files were arranged in a structure mainly based on the platform architecture introduced in the System Design Part for more convenient code management. Brief explanations for each part are as follows.

```
----------------------------------------------------------------
# Smart Contracts
contracts/
├── Library.sol     # Main library contract for book management
├── Lock.sol        # Contract for locking mechanism
----------------------------------------------------------------
# Frontend Components
src/
├── components/
│   ├── BlockchainEvent.js    # Blockchain event listener
│   ├── BookSearch.js         # Book search interface
│   ├── BookUpload.js         # Book upload functionality
│   └── ErrorBoundary.js      # Error handling component
├── lib/
│   └── utils.js              # Utility functions
├── App.js                    # Main React component
├── index.js                  # Application entry point
└── ipfs.js                   # IPFS configuration and setup
----------------------------------------------------------------
# Database and Storage
database/
└── orbitdb.mjs     # OrbitDB configuration and operations
orbitdb/            # OrbitDB data directories
```

[1] Solidity: https://soliditylang.org/

[2] Hardhat: https://hardhat.org/

[3] MetaMask: https://metamask.io/zh-CN/

```
└── [peer-ids]/    # Peer-specific storage
    ├── cache/     # Local cache
    └── keystore/  # Cryptographic keys
----------------------------------------------------------------
# Blockchain Deployment
migrations/
└── 2_deploy_contract.js   # Contract deployment scripts
ignition/
└── modules/
    └── Lock.js            # Deployment modules
artifacts/                 # Compiled contract artifacts
└── contracts/
    ├── Library.sol/
    └── Lock.sol/
----------------------------------------------------------------
```

The rest of the files are mainly for configurations, which include development and framework settings.

5 Project Result & Discussion

Our system overcomes key limitations of existing decentralized storage models by introducing several critical enhancements. Unlike IPFS, which relies solely on Content Identifiers (CIDs) for file retrieval, we integrate OrbitDB to enable intuitive metadata-based searches, making file discovery accessible to non-technical users. By storing only metadata in OrbitDB while keeping file contents in IPFS, our design remains lightweight, ensuring faster searches and retrievals.

Transparency and accountability are achieved through blockchain audit logs, which record all user actions immutably. This ensures that only file owners can delete their uploads, safeguarding data integrity. Additionally, a transaction-based ETH payment system regulates resource usage, preventing abuse and promoting fairness in shared storage.

According to our research, these improvements position our system as a more usable, efficient, and transparent alternative to existing solutions, as summarized in Table 1.

Feature	**IPFS**	**OrbitDB**	**Storj**	**Our Project**
Decentralized Storage	✓	✓	✓	✓
Content Addressing	✓	✗	✗	✓
Lightweight Metadata-based Search	✗	✓	✗	✓
Transparent Governance	✗	✗	✓	✓
Incentivized Usage Control	✗	✗	✓	✓

Table 1: Comparison of Features across Solutions

6 Conclusion

By improving the searchability of IPFS files through OrbitDB, integrating blockchain for transparent user behavior tracking, and leveraging virtual currency to regulate system usage, our project offers a practical and scalable solution to the challenges of decentralized data storage. This combination of technologies offers the promise of more secure, transparent, and efficient data storage systems in the future, paving the way for more widespread adoption of Web3 technologies.

Acknowledgment

This project is the final project of course COMP4651, HKUST. We have spent a month building our GitHub repository since November 3rd. You are welcome to view our development progress through this link.

We acknowledge the use of generative AI tools for linguistic refinement of certain sections of this document. The authors retain full responsibility for the content and its accuracy.

We thank the support from Prof. Guo and all the TAs of COMP4651 for their valuable guidance and assistance throughout this project. Their insights and feedback were instrumental in shaping the direction and execution of our work.

References

[Ben14] Juan Benet. Ipfs - content addressed, versioned, p2p file system, 2014.

[Con24] OrbitDB Contributors. Orbitdb: Peer-to-peer databases for the decentralized web. https://github.com/orbitdb/orbitdb, 2024. Accessed: 2024-11-21.

[frnd] foundry rs. Foundry: A blazing fast, portable and modular toolkit for ethereum application development. https://github.com/foundry-rs/foundry/tree/master, n.d. Accessed: 2024-11-25.

[Lab24] Storj Labs. Storj, 2024.

[webnd] web3.js. *web3.js documentation*. Read the Docs, n.d. Accessed: 2024-11-25.

[Wik24] Wikipedia. Web3 — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Web3&oldid=1257318332, 2024. [Online; accessed 21-November-2024].

[Woo] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger.