International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 09 | Sept - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Blockchain Based Decentralized Storage Drive

Shrunkhla Wankhede[1], Anjaney Charmode[2], Bhushan Ambulkar [2], Satyam Urkude[2], Abhishek Bijekar[2]

[1]Assistant Professor, Department of CSE, Priyadarshini Bhagwati College of Engineering
[2]Students, Department of CSE, Priyadarshini Bhagwati College of Engineering

---

Abstract - The "Decentralized Storage Drive" project introduces a secure, blockchain-based platform for decentralized data storage and sharing. By integrating the Interplanetary File System (IPFS) for decentralized storage, the platform ensures that data remains immutable and resistant to censorship. Solidity smart contracts deployed on the Ethereum blockchain provide robust mechanisms for access control, allowing users to manage data ownership and permissions without relying on centralized authorities. A React-based front-end interface simplifies user interaction, enabling seamless data upload and access management. This project addresses critical issues related to data privacy, security, and ownership, offering a user-friendly solution that empowers individuals with greater control over their digital assets. Throughout the development of this project, a focus on security, user control, and decentralization has been paramount. The rigorous testing of smart contracts and the integration of secure authentication methods ensure that the platform is both reliable and resilient to potential threats. The "Decentralized Storage Drive" represents a significant advancement in the field of decentralized applications, demonstrating the practical application of blockchain technology in everyday data management.

***Key Words*:** Decentralized Storage, Blockchain Technology, IPFS(InterPlanetary File System), Smart Contracts, Data Integrity, Access Control

1. INTRODUCTION

In the digital age, data management and storage have become important issues for individuals and organizations. Centralized systems, while useful, pose significant risks in terms of data privacy, security, and governance. These systems are often controlled by a single organization and are therefore vulnerable to data breaches, unauthorized access, and censorship. As the volume of sensitive data increases, the need for more secure, transparent, and user-friendly data management methods also increases. to solve these problems. The project aims to transfer data management from centralized organizations to individual users, allowing them to have full ownership and control of digital assets. Peer-to-peer file system. Unlike traditional cloud storage where data is stored on centralized servers, IPFS ensures immutability, distribution, and immutability of data. This means that once a file is transferred to IPFS, it cannot be modified or deleted anywhere, ensuring its integrity and longevity. Data ownership and management. These smart contracts operate in a self-governing manner, subject to rules governed by the data owner. Users can easily grant or revoke access to their information, ensuring that only authorized individuals can view or modify it. This decentralized approach to management eliminates the need for intermediaries, thereby increasing security and trust. This user-friendly interface allows people to easily upload files, manage files, and manage access without requiring deep knowledge of blockchain technology. Integrating IPFS and smart contracts into the front-end provides a seamless and user-friendly experience. By combining the benefits of blockchain technology, IPFS, and modern network development, the project offers effective solutions to the challenge of securing data storage and sharing. It gives users more control over their data, reduces dependency on centralized organizations, and increases overall data security. As the demand for secure and transparent data management continues to grow, the project offers future-oriented solutions based on the principles of decentralization and user empowerment.

2. PROPOSED MODEL

The proposed model uses decentralized storage and blockchain technology to provide secure, transparent and efficient information management through effective access control.

2.1 Components of the model:

- Front-end interaction: Users interact with the system through a browser-based front-end designed using HTML, CSS, JavaScript, and React. The front-end uses libraries such as Ethers.js to communicate with the back-end and smart contracts published on Ethereum.
- Metamask integration: Users log in and interact with the system using the Metamask browser extension, which acts as a wallet and personal manager. Ethereum addresses are captured in windows.ethereum, allowing the system to identify and describe users.
- IPFS file storage: User-uploaded files are stored on IPFS, and each file is assigned a unique hash value. This hash value is used as a reference to retrieve data from the decentralized IPFS network. Only data hashes are stored on the Ethereum blockchain, reducing storage costs.
- Smart Contract for Ownership and Permissions: Smart contracts written in Solidity manage file ownership, permissions, and payments.
- Cryptocurrency Payments: Ethereum is used as a currency for sharing information. Users can share information with others by sending small amounts of cryptocurrency, ensuring that only authorized users can access the content. Payments are

verified and recorded on the blockchain, and smart contracts manage secure transactions.

2.2 Functional agreement:

File upload: Users upload files via the front-end interface. Data stored in IPFS and its hash value are recorded on the Ethereum blockchain using smart contracts. The user maintains ownership of the file, and the contract tracks metadata, including the wallet owner's location and permissions. Permission required (view, edit, download). Smart contracts ensure that data hashes and shared ownership details are secure. Payment is made using Ethereum tokens. The smart contract verifies their access rights before allowing them to download files.

2.3 Advantages of this model:

Decentralization, Leveraging IPFS and Ethereum, the system eliminates the need for centralized servers and provides greater flexibility and security. Chain technology ensures immutability of data ownership and shared permissions. Users are protected from unauthorized access and transactions are verified. IPFS manages large storage in a distributed and efficient manner. Users can trust the system without the need for intermediaries. especially for privacy-sensitive and secure data transfer scenarios.

3. PROJECT ARCHITECTURE

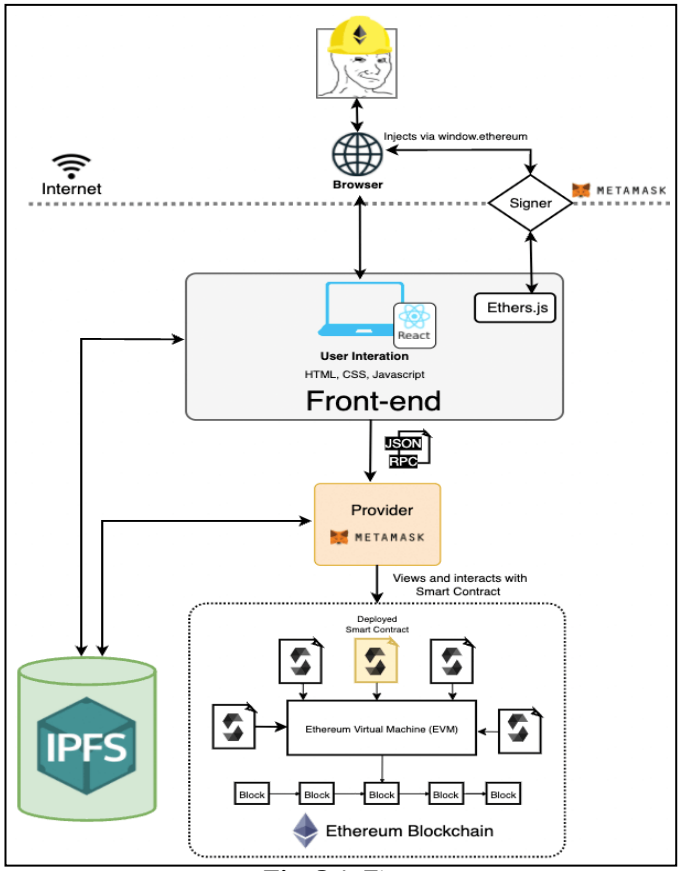

Fig -3.1: Figure

The decentralized data sharing architecture combines blockchain technology with the InterPlanetary File System (IPFS) to create a secure environment for data storage and sharing. The system is divided into several layers, providing user experience, security, transparency, and classification at every step. The front-end applications, built using technologies like HTML, CSS, JavaScript, and React, serve as gateways for users to interact with the decentralized system, allowing them to perform tasks such as uploading, sharing, and accessing information.

The front-end communicates with the blockchain using Ethers.js, a JavaScript library that handles operations like recording data uploads, sharing data, or checking permissions. The costs associated with blockchain storage are high, so stored data is sent to IPFS, a decentralized storage space that breaks data into smaller pieces, distributes them across nodes, and generates unique hash points for each archive. This model ensures that only the hash value of the data is stored on the Ethereum blockchain, while the content of the data is stored on IPFS nodes for resource efficiency.

Smart contracts, written in Solidity, form the basis for managing ownership information, permissions, and cryptocurrency payments. These contracts include data uploading, sharing, governance, and other functions. When a user uploads a file, the contract collects the hash value and associates it with the user's Ethereum address. Users can set permissions in the smart contract and specify which addresses have permission to view or download the data.

The Ethereum Virtual Machine (EVM) hosts smart contracts, verifying the authenticity and correctness of all transactions. Blockchain creates an immutable system of transactions by keeping records of all transactions in blocks, providing transparency and trust to users. The architecture aims to share data, eliminate dependency on centralized sources, and give users control over their data.

4. IMPLEMENTATION

Implementing shared data integration brings the proposals to life with carefully designed steps that enable user interaction, primarily front-end applications, the IPFS network, and the Ethereum blockchain. This project uses modern business development technology, blockchain infrastructure, and shared storage to create a secure, transparent, and shared data and storage system. The frontend interacts with the Metamask browser extension, which acts as a bridge between web applications and the Ethereum blockchain. Metamask allows users to manage their Ethereum wallets and facilitate transactions, allowing them to interact with smart contracts directly from their browser. Instead of being stored on the blockchain, which would be prohibitively expensive and inefficient, data will be transferred to the InterPlanetary File System (IPFS). IPFS is a distributed storage system that breaks files into smaller pieces, stores them on a network of nodes, and creates hashes that mark files as unique. This hash value is important because it acts as a reference for information in a decentralized network, allowing users to store information without the need for a central server. Smart contracts manage ownership of data, control access, and payments for sharing data. When a user uploads a file, the frontend communicates with the Ethereum blockchain via Ethers.js, a library that facilitates interaction with the blockchain. The smart contract writes a hash of the archive along with the users Ethereum wallet address to verify ownership of the archive. This provides transparent and immutable information on file uploads,

allowing its members to be verified to the user on the blockchain, even if the data is stored on IPFS. Users can choose to share data with other Ethereum addresses by setting permissions in the smart contract. The frontend allows users to set these parameters, and once approved, the changes are sent to the blockchain, where smart contracts update the rules for the data. The recipient can retrieve the data by interacting with a smart contract that verifies access rights and then allows IPFS to retrieve the hash of the data. IPFS uses content-based storage, ensuring that the data stored using the hash is the first data stored. The decentralized nature of IPFS also ensures that files are available as long as they are hosted on at least one node in the network. Every interaction, whether it's uploading a file, sharing a file, or paying for access, is recorded in the blockchain database, creating a proof, and there is no proof of work. Build and deploy: Tools like Remix IDE or Truffle combined with a blockchain development environment like Ganache provide an easy way to write, test, and deploy smart contracts locally and on the live Ethereum network. By ensuring that all interactions between the front-end, IPFS, and the blockchain are seamless and efficient. For example, ensuring that IPFS nodes are still available to store and store data, improving the fuel price for transactions on the Ethereum network, and preventing interactions with smart contracts.

5. RESULT

The system uses the InterPlanetary File System (IPFS) for distributed file storage to ensure the security and distribution of data. Each file is assigned a unique hash code that allows efficient access and ensures data integrity. Users can connect to their Ethereum wallets via MetaMask, allowing them to interact with business applications, including trading platforms. Reduce the risk of unauthorized access. Users can share information by paying in Ethereum, which adds a layer of security by limiting access to authorized users. This approach prevents unauthorized transfer of information and transactions, thereby increasing the overall security and reliability of information sharing. The combination of IPFS and blockchain creates a secure environment for users while increasing transparency and trust in digital assets.

5.1 Sign in with Metamask

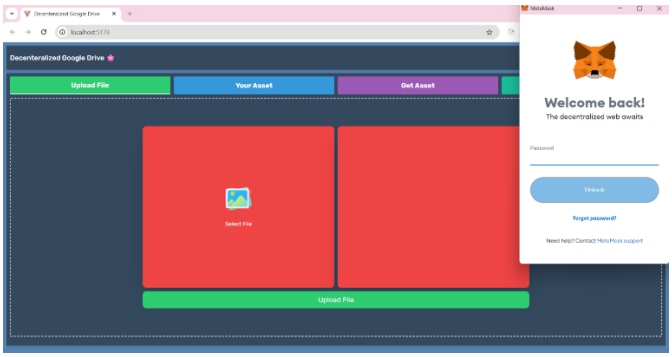

Fig -5.1: Landing Page

MetaMask is a browser-based cryptocurrency wallet that allows users to interact with transactions on the Ethereum blockchain and other blockchain networks. The "Sign Up with MetaMask" feature provides security and user authentication, making it ideal for services like Google Drive sharing. Users install MetaMask as a browser extension or mobile app and connect their wallet to the app. Users authenticate by signing encrypted messages with their private keys stored in MetaMask, allowing apps to verify user ownership. Functions include authentication, transfer, and data management. No sensitive information is stored in the app, and additional features are only accessible to public wallet addresses. Users must always use a username and password when logging in, reducing the risk of phishing attacks, password theft, or data breaches. Keys and digital tokens. The authentication process is controlled by the blockchain, which provides safer and more secure authentication. The system uses IPFS to cryptographically hash data to ensure data integrity and frequently uses smart contracts to help secure and distribute this data.

5.2 Upload a file:

Fig -5.2: File Uploading

The "file upload" process in decentralized systems like Google Drive alternatives allows users to upload and store files securely using technologies like IPFS and blockchain. Upload files. They can browse local files and select files to upload. The system usually supports many file types, including documents, images, and videos, allowing users to select compatible files. Some systems may also provide drag-and-drop functionality that allows users to easily upload files by dragging them into the application interface. Archive Name indicates the name of the selected file, while Archive Size indicates its size in kilobytes or megabytes, which is important for understanding any size limits or associated costs. File Type Confirm the type of file being uploaded (such as PDF or .jpg). In some cases, a cryptographic hash value of the data is generated as a unique identifier to ensure the integrity of the data. To ensure confidentiality and security during storage and retrieval, data can be encrypted before uploading. It is then placed in a shared storage volume such as IPFS, where it is broken into smaller pieces, distributed among nodes, and given a unique value for recovery. A link or content identifier (CID) based on the IPFS file hash, by which users can access or share files.

5.3 View Uploaded Files:

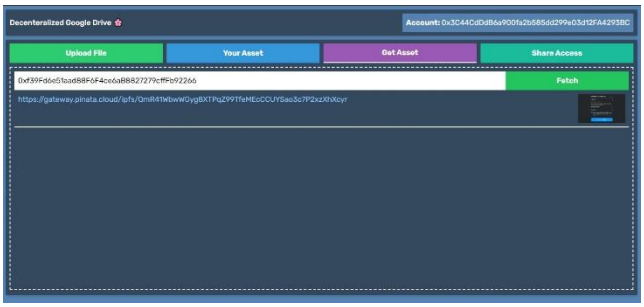

Fig -5.3: Uploaded Files view

In a decentralized system, once the data is uploaded, users can view and manage their assets through an interface that displays all the uploaded data. The process begins with the Product Details section, where information about each customer is displayed. This link provides an overview of the files previously stored in a storage location such as IPFS. Each asset in the inventory usually contains important information, including the name of the exported file, file size (in kilobytes or megabytes), file type (e.g. .jpg, .pdf, .mp4), upload instructions, and time of day. This CID serves as the "address" information of the network. provisioning. There are also various buttons next to each device that allow users to preview files, download files back to their device, generate a link or QR code for others to access, or permanently delete information. Data storage in decentralized systems relies on content-based protocols such as IPFS, where data is accessed using unique hash values (CIDs). Unlike cloud-based systems that seek information from physical locations, decentralized systems retrieve information based on its content. The retrieval process will find pieces of information scattered across the internet and reassemble them for the user. If data is encrypted for personal reasons, the user or authorized person must have the decryption key to access the content, and once this key is returned, it will be decrypted locally. Look ahead. The efficiency of data storage can vary depending on network connectivity and file size, but shared storage like IPFS is designed to facilitate faster access by achieving information close to the quality of this process. Manage and store your information in a central location.

5.4 Share Access:

Fig -5.4: Share Access

In a decentralized storage system, shared access to information is securely controlled and authorized, typically using blockchain technology and smart contracts. This ensures that the sharing process is transparent, secure, and auditable. This can include direct sharing or selecting specific users for access. When shared, the owner of the file typically defines the type of access allowed. Most options include read-only access, which allows users to view or download files, but not modify or delete them. New and full access allows users to update or upload new versions of the file, delete, or add. Public sharing allows the owner to create a link that anyone can access, while private sharing restricts access to specific individuals by sharing a unique data identifier (hash) and determining permissions based on the user's identity (e.g. wallet address (body) same as Ethereum. If the data is encrypted for privacy reasons, the owner can share the decryption key with the recipient. Without this key, even if someone accesses the location where the data is stored, they will not be able to view or download its contents. responsibility and permission. These personalized contracts write the terms of the contract directly into the code running on the blockchain. When data owners share access rights, a smart contract will emerge that records the access rights on the blockchain to ensure the process is decentralized, transparent, and tamper-proof. The smart contract defines the conditions under which the recipient can access the data, such as the type of access (read, edit, etc.) and any time limits for sharing. The data cannot be modified or audited, it specifies the content entered into the data, when access is possible, and under what conditions it is allowed. The owner of the document determines the access permission in the application, and when the smart contract executes, the access control list is updated with the recipient's wallet address and permission methods. When the recipient attempts to access the data, the smart contract verifies their authorization. If permission is granted, the recipient can download or view the information according to the permission. Enabling trusted sharing by eliminating the need for an intermediary or central authority to manage access to transactions, blockchain and smart contracts are managed authorization. This process also provides transparency and security, as all transactions related to authorization and access to data are recorded on the blockchain, preventing unauthorized access and allowing data owners to manage their data. Smart contracts can also reduce the need for manual intervention by performing defined tasks such as granting or revoking access rights. Security, transparency and efficiency.

3. CONCLUSIONS

The "Decentralized Storage Drive" project is an important step in providing secure data management solutions in an increasingly digital world. The project addresses the fundamental issues of data privacy, security, and ownership that exist in a centralized system by leveraging blockchain technology. Storage solutions that leverage immutability ensure that their data is not managed by a single source. This decentralization provides users with greater security and control by reducing the risks associated with data leakage, censorship, and unauthorized access. The platform is further enhanced by enabling strong access control processes. These

smart contracts ensure that only authorized users can access or modify data, and permissions can be granted and revoked seamlessly. This creates trust in the system as users can manage their data without relying on intermediaries on the platform. Users can send data, manage access rights, and interact with the blockchain without understanding the complexity of the technology. This accessibility is essential for widespread use of decentralized solutions. Rigorous monitoring of smart contracts and integration of authentication mechanisms ensure that the platform is both reliable and resilient to threats. By providing a secure, decentralized, user-controlled platform, the project not only addresses current needs for data privacy and security, but also sets an example for future innovations in decentralized applications. As the digital landscape continues to evolve, such solutions will play a key role in helping individuals and organizations manage their data securely and transparently.

REFERENCES

1. Zhang, Y., & Wang, Y. (2019). "A Survey on Blockchain-based Decentralized Cloud Storage Systems." IEEE Access, 7, 103617-103629. DOI: 10.1109/ACCESS.2019.2934798.
2. Lu, Y., & Xu, J. (2020). "Blockchain-based Data Sharing in Cloud Computing." Journal of Systems Architecture, 113, 101800. DOI: 10.1016/j.sysarc.2020.101800.
3. Nakamoto, S. (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System." Bitcoin Whitepaper. Available at: https://bitcoin.org/bitcoin.pdf.
4. Hassan, S. A., & Murtaza, H. (2021). "Smart Contracts: Challenges and Opportunities." Journal of Computer Networks and Communications, 2021. DOI: 10.1155/2021/8858592.
5. Xu, X., Weber, I., & Staples, M. (2019). "Blockchain Technology in the Insurance Industry: A Review." The Geneva Papers on Risk and Insurance - Issues and Practice, 44(3), 461-485. DOI: 10.1057/s41288-018-00128-7.
6. Kumar, A., & Singh, A. (2020). "Decentralized Cloud Storage: A Review." Future Generation Computer Systems, 108, 778-788. DOI: 10.1016/j.future.2019.12.052.
7. Hwang, T., & Kim, H. (2019). "Decentralized File Sharing Based on Blockchain and IPFS." Journal of Computer Virology and Hacking Techniques, 15(2), 131-139. DOI: 10.1007/s11416-018-0333-6.
8. Zhao, H., & Wang, Y. (2020). "An Efficient Data Access Control Scheme for Cloud Storage." IEEE Transactions on Cloud Computing, 8(4), 1016-1029. DOI: 10.1109/TCC.2019.2907205.
9. Guo, Y., & Ma, X. (2021). "A Review of Blockchain-based File Sharing Systems." IEEE Transactions on Network and Service Management, 18(1), 54-66. DOI: 10.1109/TNSM.2020.3038039.
10. Chen, L., & Xu, H. (2018). "Blockchain and Smart Contract for Cyber-Physical Systems: A Review." IEEE Internet of Things Journal, 6(4), 5895-5907. DOI: 10.1109/JIOT.2018.2871079.