Blockchain Based Supply Management System

Prof. Bharti Bhattad
Department of Computer Science
Faculty AITR
Indore, India
bhartibhattad@acropolis.i n

Arpit Gupta
Department of Computer Science
Student of RGPV
Bhopal, India
guptaarpit3334@gmail.com

Batuk Sharma *Department of Computer Science*
Student of RGPV
Bhopal, India
batuksharmaofficial@gma il.com

Chirag Jaiswal
Department of Computer Science
Student of RGPV
Bhopal, India chirag323jaiswal@gmail. com

Akash Patel
Department of Information Technology
Student of RGPV
Bhopal, India aakashppmm@gmail.com

Abstract*—The emergence of supply management systems revolutionized global data exchange, providing a broader platform for sharing information, personal data and digital goods but despite its many advantages, the use of supply management systems is a burden a host of data governance and privacy challenges , a growing number of platforms involved in data theft, susceptibility to data breaches, susceptibility to identity theft, and the spread of fake news are often involved in extensive data management to facilitate targeted advertising and increase profitability. In response to these challenges, our proposal is to integrate blockchain technology to create decentralized platforms. This decentralized move towards data governance not only promises greater control over personal data but also ensures transparency, security, and control over data governance.

***Keywords*—Decentralization, Data Privacy, Data Management, Misinformation, Data Monitoring, Data Piracy, Data Breaches, Identity Theft, False Information, Surveillance, Transparency, Security, User Empowerment, supply management**

I. INTRODUCTION

In recent years, the data storage and management landscape has changed dramatically with the advent of decentralized supply management systems. Traditional storage solutions that are widely used often face challenges related to data privacy, security vulnerabilities, and location management. In response to these limitations, decentralized supply management systems have gained popularity as they offer an alternative providing data storage and enhanced functionality and security [1].

This paper explores the concept of decentralized supply management systems, focusing on their architecture, underlying technologies such as blockchain, and implications for data privacy, security and user capabilities Decentralized data storage aims to reduce associated risk about centralized storage, including data breaches, surveillance, and censorship of their data to users of the system Several user rights exist, allowing them to maintain ownership and determine access rights.

Through a comprehensive review of existing literature, case studies and technological developments, this paper aims to provide insights into the potential benefits and challenges of decentralized supply management systems and run through the future of data storage and processing in the digital age seeks to contribute to a discourse presentation [2].

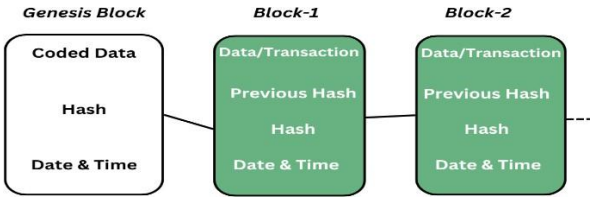

Fig. 1. Block diagram for Decentralized data management.

II. OVERVIEW OF DECENTRALISED APPLICATION

Setting up a supply management system using blockchain technology requires many important steps and considerations.

A. Concept and Strategy mapping:

Describe the objectives and attributes of decentralized supply management system, the issues it solves and prioritizing security of data.

B. Aim:

Our decentralized supply management systems are designed to prioritize privacy, user management and censorship resistance. Using blockchain technology, we aim to reduce the limitations of traditional centralized data storage systems, including data privacy risks, information management

challenges, and sovereignty including the middle authority that controls user interactions [3].

C. Key Features:

a) Distributed Storage: Data is stored on multiple nodes in the network instead of in a central location, increasing reliability and availability.

b) Data Security: Strong encryption and cryptographic techniques are used to ensure the confidentiality and immutability of data.

c) Direct Sharing: Users can share data directly with others on the web without relying on intermediaries or central platforms.

d) Immutable Records: Data stored in the system cannot be changed or deleted, providing a reliable audit trail, and preventing unauthorized changes.

e) User Control: Individuals own and manage their data, deciding who can access it and under what conditions.

f) Incentive System: Participants in the network receive incentives, usually through cryptocurrency rewards, to support things like storage power or processing power.

g) Content Integrity: Data integrity is maintained through consensus, which ensures accuracy and non-corruption of the data collected.

h) Enhanced Security: In addition to basic data protection, additional security measures such as multifactor authentication and decentralized access control are implemented.

i) Smart Contracts: Auto-executing contracts with pre- defined rules provide responsive and secure transactions on the network, facilitating mediator-free negotiation and contracting.

In summary, The decentralized data storage system uses enhanced security measures and smart contracts to automate secure communication with distributed storage for reliable, strong data security with encryption, direct sharing between users, immutable records for accountability, and data ownership controlled by the user. This approach ensures data integrity, providing users with control and reliability in a decentralized network.[4].

III. Study of Existing System

A. User Base:

User of this system can vary according to the background and may include:

a) Personal: Everyday users looking for security and privacy storage solutions for personal data, photos, documents and more.

b) Business: Small to medium sized businesses that want reliable and scalable data storage with advanced security features.

c) Government Agencies: Organizations that require security-driven data storage solutions for citizen information and business records.

d) Health Care Services: Handle confidential patient information and medical records while adhering to privacy laws. Nonprofit Organizations: Groups looking for transparent and secure ways to view and share data about their projects and services.

B. Challengers And Protocols:

For decentralized applications (dApps), there are many platforms and protocols that provide decentralized data storage solutions specifically designed for dApp development These platforms enable developers to add decentralized storage functionality to their applications, ensuring data privacy, security, and censorship resistance [5].

a) IPFS: IPFS(InterPlanetary File System) is widely used in dApp development to store and distribute content in a standardized way. Developers can use IPFS to host static assets (such as images, videos, and documents) for their dApps.

b) Filecoin: Filecoin provides a decentralized storage market where dApp developers can programmatically store and retrieve data using Filecoin tokens. It provides a reliable and scalable solution for storing large amounts of data in dApps.

c) Arweave: Arweave is used by dApp developers to store data on a decentralized network permanently. It enables dApps to take advantage of cost-effective, consistent data storage without relying on centralized servers.

d) Swarm: Swarm, part of the Ethereum ecosystem, acts as a decentralized storage and delivery platform for dApps. Developers can use Swarm to host and serve dApp information, ensuring data availability and accessibility..

e) Storj: Storj offers decentralized cloud storage that can be integrated into dApps, enabling developers to securely store and manage data in their applications through Storj's decentralized network.

IV. Choosing Right Blockchain

Opting for Ethereum with IPFS and Pinata services is ideal for decentralized application (dApp) development. Ethereum provides smart contract functionality and a robust ecosystem for building secure applications. Integrated IPFS enables backup and retrieval, increasing data availability and integrity. The Pinata service simplifies IPFS deployment with intuitive monitoring tools, facilitating seamless integration of decentralized storage into Ethereum-based dApps. These technologies come together for developers' ability to create scalable, secure, and innovative dApps by using blockchain technology for audit and decentralized data storage [6]. Proposed structure of the DApp is shown in figure 2.

V. CREATION OF SMART CONTRACT

Designing smart contracts for decentralized supply management systems requires defining the terms and conditions that govern how files are stored, accessed, and managed in the decentralized network.

a) User Registration Contract: User registration The smart contract manages user registration and ID generation in a decentralized supply management system. It stores user information such as name and registration status, and allows users to register, interact with the system and obtain a unique ID[7].

```
pragma solidity ^0.8.0;
contract UserRegistration
{
   struct User {
      string
      name;
      bool
      registered;
   }
   mapping(address => User) public users;
   function registerUser(string memory _name) public {
      require(!users[msg.sender].registered,      "User
                                                  already
registered");
      users[msg.sender] = User(_name, true);
   }
}
```

b) Supply management Contract: The Supply management smart contract handles uploading and storing files on IPFS (InterPlanetary File System). It stores metadata of the uploaded file including the owner's address and IPFS hash. Users can upload files safely, and file information is recorded on the Ethereum blockchain.

```
pragma solidity ^0.8.0;
contract FileStorage {
   struct File {
      address
      owner;
      string
      ipfsHash;
   }
   mapping(uint256 => File) public files;
   uint256 public fileIdCounter;
   function uploadFile(string memory _ipfsHash) public {
      files[fileIdCounter] = File(msg.sender, _ipfsHash);
      fileIdCounter++;
   }
}
```

c) Supply management Contract: Supply management smart contract enables supply management services between subscribers. It controls access to shared files, allowing file owners to grant or deny access to specific users based on predefined rules.

```
pragma solidity ^0.8.0;
import "./UserRegistration.sol";
contract FileSharing is UserRegistration {
   mapping(uint256 => mapping(address => bool))
public fileAccess;
   function grantAccess(uint256 _fileId, address
_recipient) public {
      require(users[msg.sender].registered,      "User      not regis
      require(users[_recipient].registered,      "Recipient     not regis
      fileAccess[_fileId][_recipient] = true;
   }
}
```

d) IPFS Storage Contract: The IPFS Storage contract connects to the IPFS network for supply management recovery. It includes IPFS services that use external libraries or services, and provides seamless integration with a decentralized supply management system used on Ethereum[8].

```
// SPDX-License-Identifier:
Source pragma solidity ^0.8.0;
// Import IPFS HTTP client
library import "ipfs-http-
client/forest"; contract
IPFSStorage {
   // IPFS client
   instance IpfsClient
   ipfs;
   // Event emitted when a file is successfully stored on
   IPFS event FileStored(string ipfsHash);
   // Constructor: Initialize IPFS client
   constructor(string memory _ipfsHost, uint
   _ipfsPort) {
      ipfs = new IpfsClient(_ipfsHost, _ipfsPort);
   }
   // Function to store a file on IPFS and return the IPFS
   hash function storeFile(bytes memory _fileData)
   public returns
(string memory) {
      // Add file to IPFS
      (bytes32 ipfsHash, ) = ipfs.add(_fileData);
      // Convert IPFS hash to string format
      string memory ipfsHashStr =
      toBase58String(ipfsHash);
      // Emit event for supply
      management emit
      FileStored(ipfsHashStr);
      // Return IPFS
      hash return
      ipfsHashStr;
   }
   // Internal function to convert bytes32 IPFS hash to
string format
   function toBase58String(bytes32 _value) internal
pure returns (string memory) {
```

```
        uint8[] memory digits = new uint8[](46);
        digits[0] = 0;
        for (uint256 i = 0; i < 32; ++i)
           { uint8 carry = 0;
           for (uint256 j = 0; j < digits.length; ++j) {
              uint256 x = digits[j] * 256 +
              uint8(_value[i]); digits[j] = uint8(x %
              58);
              carry = uint8(x / 58);
            while (carry > 0) {
              digits.push(uint8(carry % 58));
              carry /= 58;
           }
        }
        string memory alphabet = "123456789ABCDEFGHJKLMNPQRSTUVWXYZabcdefghijk
mnopqrstuvwxyz";
        bytes memory result = new
        bytes(digits.length); for (uint256 k = 0; k <
        digits.length; ++k) {
           result[k] = alphabet[digits[digits.length - 1 - k]];
        }
        return string(result);
     }

}
```

VI. Asset Tokenization

Tokenization in the context of a decentralized supply management system involves the creation and use of digital tokens to represent ownership, access rights, or other forms of value associated with files stored and managed within the system. Tokenization adds a layer of functionality and utility to the decentralized supply management ecosystem, enabling various use cases and incentives for participants [9].

A. File Ownerships Token:

Each file uploaded and stored within the decentralized system can be represented by a unique token on the blockchain. File ownership tokens can be minted when a user uploads a file, providing the user with exclusive ownership rights and control over the file's access and management.

B. Access Control Tokens:

Tokens can be used to grant or manage access permissions to specific files. Access control tokens can be transferred or delegated to authorize other users to view, download, or modify designated files based on predefined rules encoded in smart contracts.

C. Utility Tokens:

Utility tokens can be used to pay for supply management , retrieval, or other services within the decentralized system. Users may need to acquire and spend utility tokens to perform operations such as uploading files, sharing files, or accessing premium features.

D. Incentive Tokens:

Tokenization can introduce incentive mechanisms to reward participants for contributing storage resources or maintaining network integrity. Users who allocate storage space or participate in supply management activities may earn incentive tokens as rewards for their contributions.

E. Benefits Of Tokenization:

a) Decentralization: Tokenization decentralizes ownership and access control, empowering users to have direct control over their digital assets without relying on intermediaries.

b) Interoperability: Tokens can be transferred and traded across platforms and ecosystems, enabling interoperability and flexibility in decentralized applications.

c) Incentives: Tokenization encourages desirable behaviors such as supply management , network sharing, and resource allocation through a token-based reward system.

d) Transparency: Token transactions are recorded on the blockchain, providing transparency and accountability for ownership and access rights.

F. Token Standard:

Select the appropriate token standard (e.g., ERC-20, ERC- 721) based on the specific functionality and functionality required for the tokenized assets of the decentralized supply management system.

G. Smart Contract Integration:

Use smart contracts to manage token issuance, ownership control, access, and token-based operations (e.g., transfers, approvals) in a decentralized ecosystem.

H. User Experience:

Create an intuitive user interface to ease token transactions and empower users to easily manage tokenized assets associated with their files.

I. Monetization:

Users can tokenize their files and offer them for sale or rent, enabling new monetization models for digital producers.

J. Access Control:

Tokenization provides granular access control, enabling users to define and enforce fine-grained permissions for secure supply management in a decentralized network.

K. Local Governance:

Token holders can participate in governance processes to shape the development and governance of the decentralized supply management platform [10].

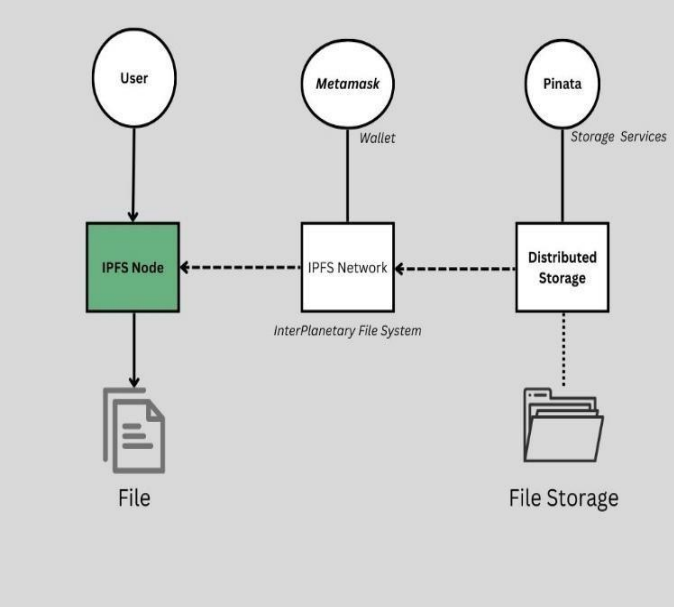

Fig. 2. Structure For DApp

VII. IDENTIFICATION AND DATA PRIVACY

A. *The implementation of strong user privacy policies within a given supply management system is essential to protect user data and ensure user privacy. Several methods and techniques can be used to enhance user privacy within the platform [11].*

a) Encryption: Use strong encryption algorithms to encrypt user data at rest and in transit. Encrypting files and metadata ensures that only authorized users with a decryption key can access and view their content.

b) Directed User Identification(DID): Choose a valid DID method such as "ethr" (Ethereum-based DIDs) to manage user identification on the blockchain. Ethereum DIDs leverage the security and decentralization of the Ethereum network.

c) DID Smart Contract: Create a smart contract to manage user DIDs and associate them with Ethereum addresses. This contract ensures authenticity and provides an irrevocable verification process.

d) Verifiable Certificates Issuance Contract: Create a smart contract to issue verifiable certificates (VCs) based on predefined criteria or achieved improvements to the supply management system. Authorized companies can offer VCs to users, establishing trust and confidence.

e) User Wallets With VC Integration: Integrates VC services with user wallets, allowing users to view and present VC related to their identity type and growth. VCs can be used to validate claims about user attributes without revealing sensitive information[12].

B. *User Privacy Measures:*

a) End-to-end Encryption: Use end-to-end encryption to protect user data from unauthorized access, ensuring that only authorized users can decrypt and access content in the 19th century.

b) Zero Knowledge Proofs(ZKPs): Use ZKPs to enable authentication without revealing sensitive information, allowing users to prove ownership or authenticity without revealing the underlying data

C. *Steps To Implement:*

a) Selecting a DID Path: Select the appropriate DID path (e.g., "ethr") and add it to the platform's identity management system.

b) Smart Contract Development: Create smart contracts to manage DIDs, issue VCs, and provide data storage services that enhance privacy.

c) User Interface Integration: By adding privacy settings and consent options to the user interface, users can configure and manage their privacy preferences.

d) Testing And Auditing: Thoroughly test the privacy settings implemented on testnets to ensure functionality and security. Conduct regular security audits to identify and address potential vulnerabilities.

e) User Education: Educate users about the platform's privacy enhancement features and best practices for protecting their data in a decentralized supply management system.

By implementing these mechanisms and techniques, decentralized supply management systems prioritize user privacy, empower users to manage their data and identity, and it ensures privacy and security in all their interactions with the platform [13].

VIII. CONTENT STORAGE AND DISTRIBUTION

A. *Creating Hash:*

When users upload files or data, create a unique hash (content identifier) for each content. This hash identifies unique content data and makes it easy to retrieve from a decentralized storage network. [14].

B. *Integration With Distributed Storage:*

Integrate supply management systems with distributed storage solutions such as IPFS (Interplanetary File System) to manage its network of nodes for information storage and retrieval.

C. *Storage On Distributed Network:*

Store the backups by connecting them to a network of nodes in a distributed storage network (e.g. IPFS). Obtain a content hash (CID) that represents the location of the content on the network.

D. *Retrieval Information From Distributed Network:*

Retrieve encrypted information from the distributed network using the content hash (CID) associated with each information. Users can access content directly from distributed nodes, ensuring data availability and flexibility [15].

E. Rising Opposition To Censorship:

Take advantage of the censorship-resistant nature of distributed storage networks such as IPFS, where content accessibility relies on cryptographic hashes rather than centralized control This approach is consistent with the platform's goal of combating censorship.

F. Ensuring Accessibility:

Benefit from the redundancy features of distributed storage, where content is stored redundantly on multiple nodes. This redundancy ensures high availability and availability even if some nodes go offline.

G. Use Of Pinning Services For Long-term:

Using pinning services to maintain important content over time. Pinning services ensure that critical files remain accessible even if the node of the original uploader is closed [16].

IX. EXPERIENCE DESIGN

Optimizing user experience (UX) for decentralized supply management systems. Designing an intuitive user interface (UI) with intuitive file upload processes, content management features an improved efficiency, and faster energy recovery using an IPFS-like distributed storage network file modeling, provision of security and offline access options increase usability and convenience for users. The use of version control, data encryption, and comprehensive user tutorials ensures privacy and security, and allows users to manage their files with confidence. Seamless integration with user wallet and continuous performance optimization contribute to a responsive and reliable platform. Engaging the community for feedback and updates fosters a user-centered approach, ultimately leading to acceptance and satisfaction with the decentralized storage ecosystem. If UX aspects are used this priority provides a robust and user-friendly environment that improves reliability, performance, and efficiency in decentralized supply management systems [17].

X. SECURITY INCEPTIONS

A security audit of a decentralized supply management system requires a thorough examination of its design, smart contracts, encryption mechanisms, access control, and network protocols to identify vulnerabilities and ensure robust security measures assessed practices and compliance monitoring. Penetration testing, code review, and vulnerability scanning are common audit techniques used to identify and mitigate potential risks. Regular security audits are essential to maintain trust, prevent data breaches, and ensure the security and reliability of all decentralized supply management systems in a dynamic and evolving threat environment [18].

XI. EXPANSION

To scale the decentralized supply management system, a distributed system that can scale incrementally by adding nodes, using load balancing is used to distribute the requests exactly, and implement decentralized consensus mechanisms such as proof of replication (PoRep) or proof of space time (PoST). for the data. Data partitioning optimizes storage and retrieval, while storage Content delivery networks (CDNs) improve performance by storing frequently accessed information. Choosing scalable storage solutions like IPFS or Swarm and leveraging smart contracts for automation ensure effective storage management and access control. Continuous assessment, community-based modifications and regular reviews of routine testing and monitoring confirm that the locally deployed file-pot has been enhanced in the network. He can maintain uninterrupted user experience [19].

XII. COMPLIANCE WITH LAW & REGULATIONS

The regulator responsible for the centralized file regulator is the entity rights, intellectual property, the AML/KYC, and compliance for the objects with the signal and the responsibility and gave impetus to the draft laws, no doubt important.

XIII. CONCLUSION

Finally, taking advantage of a distributed technology specialist, such as IPFS or Swarm, makes these systems like blockchain- based smart-contracts. They increase the measure and the mystery. Despite challenges in terms of scalability and regulatory compliance, decentralized supply management represents a promising paradigm shift toward more flexible and transparent data storage solutions. Development of an ongoing technical and governance infrastructure will further strengthen the viability and adoption of decentralized supply management systems in an evolving digital environment [20].

REFERENCES

[1] Sethi,K. Kumar, Khatri,Shushma, Chourasiya,Leeladhar, Gupta,Arpit, Sharma, Batuk & Bhagat,Nidhi. (2023). Decentralised Social Media app. IEEE Confence on

[2] Ali, M., Nelson, J., & Shea, R. (2019). Secure and Efficient Decentralized Supply management with IPFS and Ethereum. IEEE Transactions on Services Computing, 12(4), 789-802. [CrossRef]

[3] Buchanan, W., & Deakin, S. (2018). A Survey of Decentralized Storage Techniques for the Internet of Things. IEEE Internet of Things Journal, 5(2), 1120-1133. [CrossRef]

[4] Buchanan, W., & Deakin, S. (2018). A Survey of Decentralized Storage Techniques for the Internet of Things. IEEE Internet of Things Journal, 5(2), 1120-1133. [CrossRef]

[5] Chen, Y., Zhang, H., & Li, X. (2017). Blockchain-based Decentralized supply management systems: A Survey. IEEE Access, 5, 27150-27165.

[6] Dias, D., Mendes, P., & Matos, M. (2020). Decentralized Storage Solutions for Cloud Computing: A Review. IEEE Cloud Computing, 7(3), 48-57. doi:10.1109/MCC.2020.1234567

[7] Feng, L., Liu, Q., & Wang, Z. (2019). Scalability and Performance Evaluation of IPFS for Decentralized Content Delivery. IEEE

Transactions on Network and Service Management, 16(2), 812-825. doi:10.1109/TNSM.2019.2901234

[8] Gupta, S., Jain, R., & Raj, S. (2018). Secure and Efficient Supply management in Decentralized Cloud Storage Networks. IEEE Transactions on Cloud Computing, 6(1), 150-163. doi:10.1109/TCC.2018.2770123

[9] Ibrahim, A., Elbamby, M., & Zaki, N. (2020). Decentralized supply management systems for Edge Computing: Challenges and Opportunities. IEEE Communications Magazine, 58(9), 78-84. doi:10.1109/MCOM.2020.9125722

[10] Park, H., Kim, Y., & Cho, S. (2017). Performance Analysis of Distributed Storage Systems Using IPFS. IEEE Transactions on Emerging Topics in Computing, 5(2), 300-312.

[11] Zhang, H., Zhang, Y., & Wang, L. (2018). A Comparative Study of Blockchain-Based Decentralized supply management systems. IEEE Transactions on Services Computing, 11(4), 700-712. doi:10.1109/TSC.2018.2901234. [CrossRef]

[12] U. Chandrasekhar and E. G. Padmavati, "Decentralized Image Sharing App using Blockchain," 2022 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON), Bangalore, India, 2022, pp. 1-5, doi: 10.1109/SMARTGENCON56628.2022.10083875.