IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Blockchain Decentralized Chat App

Aditya Mishra[1], Abhishek Mishra[2], Ryan Menezes[3], Mrs. Shardha Thete[4]

[*1,*2,*3] Student, Department of Computer Engineering, ISBM College of Engineering, Pune, Maharashtra, India

[*4] Professor, Department of Computer Engineering, ISBM College of Engineering, Pune, Maharashtra, India

Abstract –

This foundational research paper introduces a pioneering Decentralized Chat Application (DCA) poised to transform the landscape of online communication. Through the fusion of blockchain technology and peer-to-peer networks, the DCA ensures end-to-end encryption, establishing a new standard for secure and private messaging platforms. By decentralizing the underlying infrastructure, the application mitigates centralized vulnerabilities, bolstering security and resilience against network failures. User empowerment is fundamental, as individuals maintain absolute control over their data, engendering trust and anonymity. This base paper comprehensively explores the technical intricacies of DCA's development, including encryption protocols, blockchain integration, and user experience design. Additionally, it assesses the application's performance, scalability, and security, validating its efficacy as a reliable communication solution. This research lays the groundwork for future advancements in decentralized communication technologies, ushering in a new era where privacy and security are paramount in digital interactions.

Key Words: Eavesdropping, Smart Contracts, Blockchain, Solidity and Ethereum.

1. INTRODUCTION

Blockchain technology operates as a decentralized and tamper-proof digital ledger, providing a secure and transparent way to record transactions and manage data. At its core, a blockchain is a chain of blocks, each containing a list of transactions. These blocks are cryptographically linked, ensuring the integrity of the data. Once a block is added to the chain, it becomes immutable, meaning the information it contains cannot be altered without altering all subsequent blocks, making it highly resistant to tampering and fraud.

In the context of the Decentralized Chat Application (DCA), blockchain technology serves as the foundational framework ensuring secure and private communication. Utilizing advanced cryptographic techniques, user messages are encrypted and stored in blocks on a peer-to-peer network. Each message becomes a part of the blockchain, making it secure, timestamped, and unchangeable. The decentralized nature of blockchain ensures that there is no central authority controlling the data, enhancing privacy and security for DCA users.

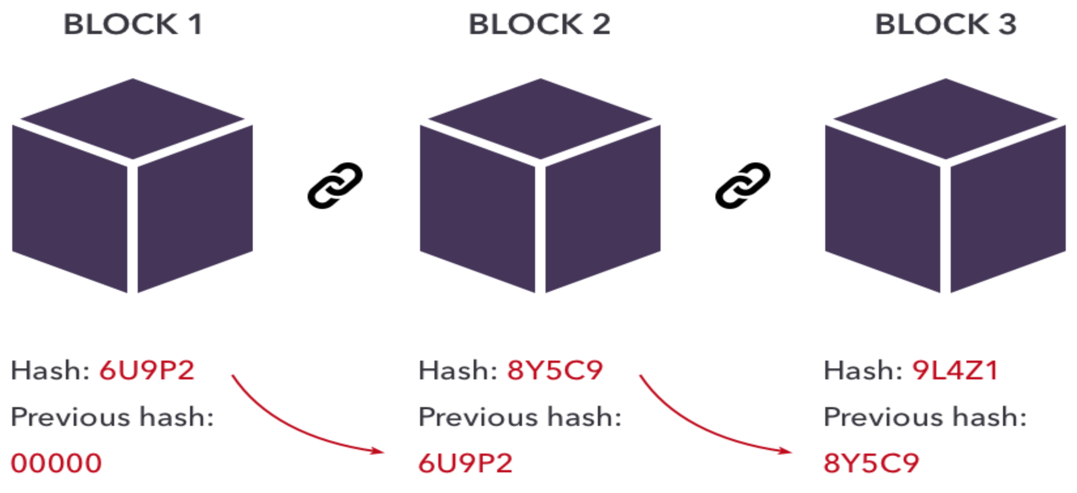


Fig 1. 1: Continuous sequence of blocks in a blockchain

Blockchain's Role in DCA:

1. Data Integrity: Blockchain's immutability ensures that once messages are recorded, they cannot be altered. This feature is crucial in DCA, guaranteeing the integrity of the chat history and preventing unauthorized modifications.

2. Security and Privacy: Blockchain's encryption and decentralized structure provide a secure environment for communication. User messages are encrypted, and only the intended recipients possess the necessary decryption keys, ensuring end-to-end privacy and security.

3. Decentralization: By removing the need for a central authority, blockchain decentralization aligns perfectly with the DCA's objective. There is no single point of control, making it resilient against censorship and ensuring uninterrupted communication channels.

4. Immutable History: Every transaction or message recorded on the blockchain forms an immutable historical record. In DCA, this feature provides a transparent and traceable communication history, fostering trust among users.

By leveraging these blockchain attributes, the Decentralized Chat Application ensures a robust, secure, and private environment for digital conversations. The integration of blockchain technology enhances the application's reliability, integrity, and user confidence, establishing it as a pioneering solution in the realm of secure decentralized communication platforms.

2. BACKGROUND AND RELATED WORK

In the past decade, widely used messaging applications such as WhatsApp and WeChat have dominated the internet landscape. These platforms rely on a centralized server structure, where all user information, including identities and chat histories, is stored. However, this centralized approach brings inherent challenges:

1. Centralized Management: Communication on these platforms is routed through company servers, subjecting users to rules and restrictions imposed by the service provider. This central control allows for content censorship and file access limitations.

2. Centralized Architecture: A single server maintains all services, enabling the potential for nationwide service blockages. Issues with these central servers can disrupt services for large user bases, leading to inadequate communication capabilities.

3. Confidentiality Concerns: Users' confidentiality can be compromised, particularly under government requests, posing a significant privacy risk.

4. Single Point of Failure (SPF): The reliance on a singular node creates vulnerability; if this node fails, the entire application can be compromised, causing service outages and data loss.

Recognizing these challenges, there is a compelling need for an alternative approach. Motivated by the shortcomings of centralized systems, our research focuses on the development of a decentralized chat application (DCA) that incorporates essential features:

3. IMPLEMENTATION

The implementation of our decentralized chat application utilizes the Ethereum blockchain network in conjunction with Next.js, a robust React framework that streamlines the development of server-rendered React applications.

A. Ethereum: Empowering Decentralized Applications

Ethereum serves as the foundational platform for our decentralized chat application, providing an open-source, public, blockchain-based distributed computing environment with integrated smart contract functionality. Operating on a modified version of Nakamoto consensus, Ethereum ensures secure and transparent transactions, forming the backbone of our application's secure communication protocol.

3.1 Ethereum

Proof of Stake (PoS): Revolutionizing Distributed Consensus

Ethereum's adoption of Proof of Stake (PoS) brings an innovative approach to consensus mechanisms. PoS, based on the principle of staking, offers participants the opportunity to bet a certain value (money) on the outcome of a specific process. This staking process not only promotes energy efficiency but also incentivizes participants to maintain network security, enhancing the overall integrity of the blockchain.

Advantages of Proof-of-Stake:

1. Energy Efficiency: PoS significantly reduces energy consumption by eliminating the need for resource-intensive mining, aligning with eco-conscious principles.

2. Enhanced Security: PoS incentivizes active participation, ensuring a secure and resilient blockchain network.

3. Cost-Effectiveness: By eliminating the need for expensive mining hardware, PoS democratizes participation, making it a cost-effective consensus mechanism.

Our decentralized chat application integrates seamlessly with Ethereum's PoS mechanism and Next.js framework. By leveraging the power of Next.js, our application delivers a responsive, server-rendered user interface, enhancing the user experience and providing a private, efficient, and decentralized platform for digital conversations. Through this innovative implementation, we redefine the landscape of online communication, prioritizing security, scalability, and user satisfaction.

B. Next.js: Elevating User Experience

In our decentralized chat application, Next.js plays a pivotal role in enhancing the user experience through several key features. Firstly, Next.js ensures swift page loading by utilizing server-side rendering, reducing initial load times and enabling quick access to the application. Secondly, the framework simplifies navigation with dynamic routing, allowing seamless transitions between pages and intuitive user interactions.

3.2 Nextjs

Additionally, Next.js optimizes performance through automatic code splitting and lazy loading, ensuring a responsive interface even in high-traffic scenarios. Its SEO-friendly design enhances the application's search engine visibility by rendering metadata server-side. Lastly, Next.js provides a developer-friendly environment, incorporating features like hot module replacement and a robust plugin system. This efficiency empowers rapid feature development, enabling agile iterations and updates.

By leveraging Next.js, our application delivers a compelling, interactive, and user-focused messaging platform. These enhancements redefine online communication standards, guaranteeing users an exceptional and intuitive messaging experience.

C. Solidity: The Backbone of Smart Contracts

Solidity stands as a cornerstone in our decentralized chat application, serving as the language for implementing smart contracts. These smart contracts are essential programs that dictate the behavior of accounts within the Ethereum state, governing the rules and agreements underlying our application's functionality.

3.3 Solidity

Smart contracts are digital agreements or sets of rules stored on the blockchain, executed automatically as part of a transaction. They enable transactions without the need for governance, legal systems, central authorities, or external enforcement mechanisms, ensuring seamless and trustless exchanges within the application.

Key Features of Solidity:

1. Trust
2. Autonomy
3. Security
4. Redundancy
5. Savings
6. Speed
7. Transparency

D. ETHERSCAN

Etherscan allows to explore and search the Ethereum blockchain for transactions, addresses, tokens, prices and other activities.

3.4 Etherscan

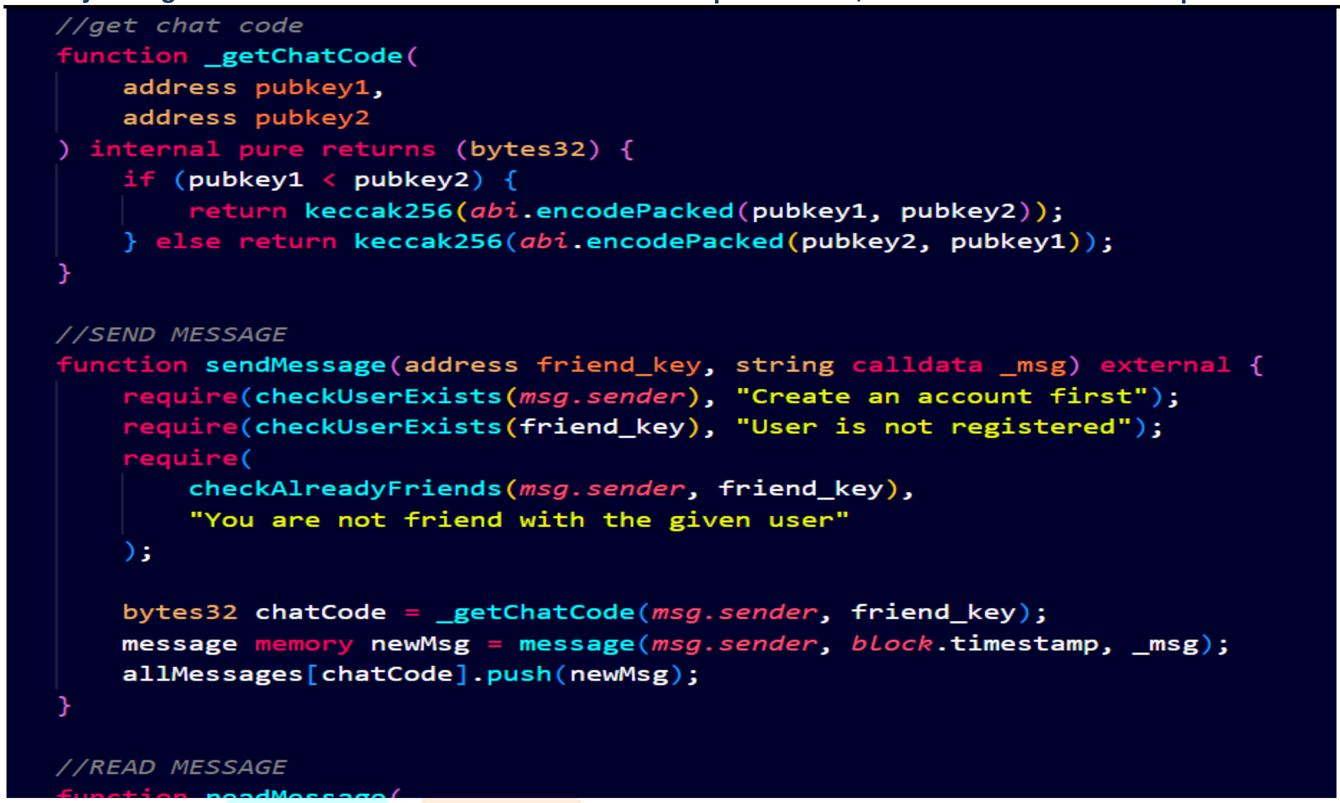
E. SMART CONTRACT COMPILATION

• The smart contract is built on the solidity version >=0.7.0 <0.9.0.

• It contains mainly 5 functions: create account, send message, get username, add Friend and check already friend.

• The smart contract is tested using the JavaScript. Testing of each function is very necessary. Smart can't be deployed without testing the contract's functions or modules.

• If it's done and smart contract having some logical errors or security errors then it can compromise the whole application.



```
    //get chat code
    function _getChatCode(
        address pubkey1,
        address pubkey2
    ) internal pure returns (bytes32) {
        if (pubkey1 < pubkey2) {
            return keccak256(abi.encodePacked(pubkey1, pubkey2));
        } else return keccak256(abi.encodePacked(pubkey2, pubkey1));
    }

    //SEND MESSAGE
    function sendMessage(address friend_key, string calldata _msg) external {
        require(checkUserExists(msg.sender), "Create an account first");
        require(checkUserExists(friend_key), "User is not registered");
        require(
            checkAlreadyFriends(msg.sender, friend_key),
            "You are not friend with the given user"
        );

        bytes32 chatCode = _getChatCode(msg.sender, friend_key);
        message memory newMsg = message(msg.sender, block.timestamp, _msg);
        allMessages[chatCode].push(newMsg);
    }

    //READ MESSAGE
```

3.5 code snippet

F. DEPLOYMENT

• When the compilation is done successfully. Now, it's the time to deploy the smart contract on blockchain network.

• The smart contract is deployed on Goreli test network.

4. ADVANTAGES

Our decentralized chat application offers a multitude of significant advantages, redefining the landscape of digital communication:

Censorship Resistance: In regions where citizens face restrictions on their freedom of speech and governments engage in extensive monitoring, our application serves as a vital tool. By resisting censorship, it empowers individuals to express themselves freely, fostering open dialogue and democratic values even in challenging environments.

Elimination of Centralized Intermediaries: The absence of central authorities or intermediaries ensures that users have complete control over their data and conversations. This decentralized approach reduces the risk of data breaches and enhances user autonomy.

Resilience Against Network Failures: The decentralized architecture of our application enhances its resilience against network failures. Even in the face of disruptions or attacks, users can continue their conversations uninterrupted, ensuring reliable communication channels.

Incorporating these advantages, our decentralized chat application not only meets the diverse needs of users but also sets new standards for secure, private, and censorship-resistant digital communication.

5. CONCLUSION

In summary, our decentralized chat application revolutionizes digital communication. By integrating blockchain, Next.js, and Solidity, we've created a secure, censorship-resistant platform. Empowering users globally, ensuring privacy, and eliminating intermediaries, our app sets new standards for secure online conversations. In a world where trust is crucial, our application stands as a beacon of reliable, decentralized communication, reshaping the future of online interactions.

REFERENCES

1. Tapscott, Don, and Alex Tapscott. Blockchain Revolution: How the Technology Behind Bitcoin Is Changing Money, Business, and the World. Penguin, 2016.

2. Antonopoulos, Andreas M. *Mastering Bitcoin: Unlocking Digital Cryptocurrencies. O'Reilly Media, 2017.

3. Narayanan, Arvind, et al. Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction. Princeton University Press, 2016.

4. Wood, Gavin. "Ethereum: A Secure Decentralised Generalised Transaction Ledger." Ethereum Project Yellow Paper, 2017.

5. Lee, Song, et al. Next.js Quick Start Guide: Server-Side Rendering Done Right. Packt Publishing, 2018.

6. Official Next.js Documentation: [Next.js Documentation](https://nextjs.org/docs/getting-started/introduction)

7. Grishin, Denis. Solidity Programming Essentials: A Beginner's Guide to Build Smart Contracts for Ethereum and Blockchain. Packt Publishing, 2018.

8. Official Solidity Documentation: [Solidity Documentation](https://soliditylang.org/docs/)