Blockchain-Secured AI AgriChat Platform

Jitesh Uikey, Pushkar Patil, Heramb Patil, Shalini Wankhade, Aman Shaikh

Information Technology Vishwakarma Institute of Information Technology, Pune, India

DOI: https://doi.org/10.51584/IJRIAS.2025.100500028

Received: 25 April 2025; Accepted: 29 April 2025; Published: 29 May 2025

ABSTRACT

Increasing fear of data privacy, security, and cen- tralization in traditional messaging applications calls for an alternative provided by decentralized applications. This paper presents the design and implementation of a decentralized chat application built on the Ethereum blockchain. Using Smart Contract in developing in Solidity, Hardhat as development framework, and MetaMask wallet in authentication methods, the application ensures safe communication that is immutable. The front-end is implemented into React.js, and interoperability with the Ethereum network for the back-end services has been facilitated using Node.js. Every communication that is sent out and received by the users is securely stored on the blockchain to prevent data interference by outsiders and ensure integrity. The result is a privacy-focused messaging platform that highlights the potential of blockchain technology to the redesign of digital com- munication. Future development aims to integrate Agricultural Assistance Chatbot for farmers, providing recommendations on sustainable techniques for farming and optimizing resource use for minimum environmental damage. This will improve the platform allowing mass users to make informed choices that will lead to productivity and sustainability in agriculture.

Keywords: Ethereum Blockchain, Smart Contracts, Solidity, MetaMask, Hardhat, Distributed Architecture, AI Chatbot

INTRODUCTION

The rapid rate of digital communications has developed many concerns on data privacy, security, and control issues related to centralized messaging systems [1]. Conventional systems are usually weak in terms of vulnerabilities regarding single points of failure, unauthorized breaches in data, and malicious access, which violates trust that should imply confidentiality and integrity of communications for the users. After rapid advancement toward the goals of increased security, transparency, and resilience in messaging solutions, decentralized applications that use blockchain to address some of these shortcomings emerged under investigation. With a de- centralized infrastructure, the control over data is redistributed back to users, thus improving security and risks associated with centralization [2].

Currently, popular chat applications like WhatsApp, Facebook Messenger, and Telegram dominate the communication landscape, serving billions of users worldwide. Even though these apps are very famous, they store and maintain their database according to central servers, and thus many hacking incidents threaten their safety. There are many examples of data breaches, hacking, and unauthorized surveillance that are increasing the sense of losing user information in such applications [3]. For instance, there are few very huge data breaches related to Telegram and WhatsApp that exposed millions of users to identity theft and unlawful exploitation of their account information [4]. These examples show some of the vulnerabilities in centralized designs, where a failure at one point can affect large numbers of users' personal information and security.

The core characteristics of blockchain technology are immutability and decentralization, which provide a good basis to build a secure messaging application [5]. One of the most widely known blockchain platforms specifically noted for its smart contract capabilities is Ethereum, as it easily enables developers to build decentralized applications with programmable logic and automatic execution [6]. Its large and active community provides a wealth of resources, tools and help. The growing ecosystem of dApps and protocols offers potential of collaboration. These factors make Ether the best choice for blockchain-based solutions.

Solidity is a curly-bracket, high-level programming language primarily for smart contracts [7]. It supports complex functions, libraries, inheritance and many features best suitable for smart contracts on the Ethereum blockchain. MetaMask is a browser extension that one utilizes to easily interact with the Ethereum blockchain. As a service, it presents itself as a digital wallet; for example, it would be possible to hold and manage your ETH tokens and interact with dApps [8]. Generally, Solidity and MetaMask go hand in hand as an efficient team for developing and launching decentralized applications on the Ethereum network. The objective of this research is to design and deploy a Decentralized Messaging Application with the Ethereum blockchain as a secure and privacy-oriented alternative to conventional messaging services. This frame-work ensures un-modifiable and tamper-proof communication by benefiting from smart contracts written in Solidity and by using user authentication through MetaMask against data leakage, unauthorized access, and vulnerability due to single points of failure weakness. Future plans will include AI-driven agricultural assistance that will enhance the functionality of the platform, thus further reflecting the broader potential of blockchain and artificial intelligence in tandem to change digital communication, such as to enable more sustainable practices.

RELATED WORK

This paper provides a comprehensive overview of consortium blockchain technology, exploring its potential to stream- line collaborative business networks. The author proposed a layered architecture and BFT-SMR (Byzantine Fault Tolerant - State Machine Replication) with a strong emphasis on the consensus mechanism's role in maintaining data integrity. Challenges inherent in consortium blockchains are discussed, thereby highlighting several open research questions for future exploration, particularly concerning scalability and governance models. The paper also touches upon various practical appli- cations and real-world uses [9], illustrating the technology's versatility across different sectors.

The paper details a blockchain-based chat application tailored for forensic investigation, addressing the critical need for secure and auditable communication within legal proceedings, and outlines its design principles based on fundamental crypto- graphic techniques and network structures. The current version supports instant messaging features, ensuring secure text-based exchanges with inherent immutability. Future development plans include the integration of voice and video communication, prioritizing robust security and privacy settings for sensitive investigative data [10].

This paper introduces a novel blockchain-oriented instant messaging framework specifically designed to mitigate security vulnerabilities prevalent in online shopping environments. The system leverages Chinese cryptography standards for enhanced security and integrates machine learning algorithms to improve user authentication accuracy and ensure message integrity. Additionally, machine learning techniques are employed for anomaly detection to bolster confidentiality and proactively identify potential security threats during online transactions [11], thus presenting a more secure approach to online communication.

This paper presents a decentralized online marketplace built on the Ethereum blockchain, aiming to overcome limitations of traditional centralized platforms such as single points of failure and data control. The application offers increased user autonomy over transactions and personal data, reduced transaction fees by eliminating intermediaries, and enhanced privacy through blockchain's inherent pseudonymity. The paper includes a thorough development process description, a performance evaluation assessing scalability and responsiveness, and a cost analysis demonstrating its economic feasibility as a viable alternative to conventional marketplaces [12]. Future work aims to integrate decentralized database solutions and optimize transaction processing efficiency.

This paper presents a network-based encrypted chat system meticulously designed to address the persistent security issues and efficiency limitations that are often embedded in most conventional chat applications. The proposed method rigorously employs examination on HTTPS (Hypertext Transfer Protocol Secure) for secure web communication, robust AES (Advanced Encryption Standard) encryption algorithms for data confidentiality, and explores other established cryptographic techniques to ensure secure information transmission. This enhanced security and efficiency is intended to be relieved with the strategic burden distribution across multiple servers through the implementation of the server farm architecture [13].

This paper focuses on significantly improving both the security robustness and operational efficiency of blockchain net- work operations through the introduction and comprehensive analysis of innovative hybrid consensus algorithms, strategically combining the inherent strengths of different established consensus mechanisms. The proposed hybrid methodology is practically demonstrated and evaluated within the specific context of the ProximaX blockchain platform, carefully highlighting the associated implementation challenges encountered and outlining promising directions for future research in the pursuit of developing more resilient, secure, and scalable decentralized network architectures [14].

This paper introduces a novel blockchain-based Industrial Internet of Things (IIoT) architecture designed for stringent security and high reliability in interconnected industrial systems. A key innovation is its multi-center, partially decentralized design, balancing autonomy and coordination. The paper details enabling technologies, data interaction, and robust security, addressing limitations of conventional IIoT architectures by leveraging blockchain’s immutability and integrity. Future work includes integrating smart contracts for automation and advanced distributed systems for enhanced remote supervision of industrial assets [15].

Table 1 presents a comparative analysis of several related works in the field. It systematically outlines the key aspects of each referenced paper, including the specific technology employed, the primary strengths highlighted by the authors, and the identified limitations of their approaches. This structured comparison allows for a clear understanding of the existing landscape of research, identifying the gaps and opportunities that motivate the current work presented in this paper.

TABLE I Comparative Analysis of Related Works

Paper	Technology	Strengths	Limitations
Ellewala et al. (2020) [10]	Ethereum, Smart Contracts	Strong encryption; decentralized architecture	Limited to text messages; no media or group chat support
Yi (2019) [11]	Chinese Cryptography, ML	Anomaly detection; message integrity with ML integration	Specific to e-commerce; not a general-purpose messaging app
Ranganthanet al. (2018) [12]	Ethereum Blockchain	Real-time evaluation; cost-effective design	Lacks dedicated messaging interface; marketplace fo- cused
Liu (2024) [13]	HTTPS, AES Encryption, Server Farm	Reliable encryption; scalable server infras- tructure	Centralized system; vulnerable to single-point failure
Our Work	Solidity, MetaMas k, React.js, Node.js	Fully decentralized; end-to-end encrypted; AI-based agricultural chatbot integration	Requires gas optimization and benchmarking; high main- net costs

PROPOSED SYSTEM

The system architecture, in Fig. 1, has several intercon- nected components that allow secure and private commu- nications to be held on the Ethereum blockchain. Diagram represents modules User Device, Backend Server, Blockchain Network and optional service Inter-Planetary File System (IPFS).

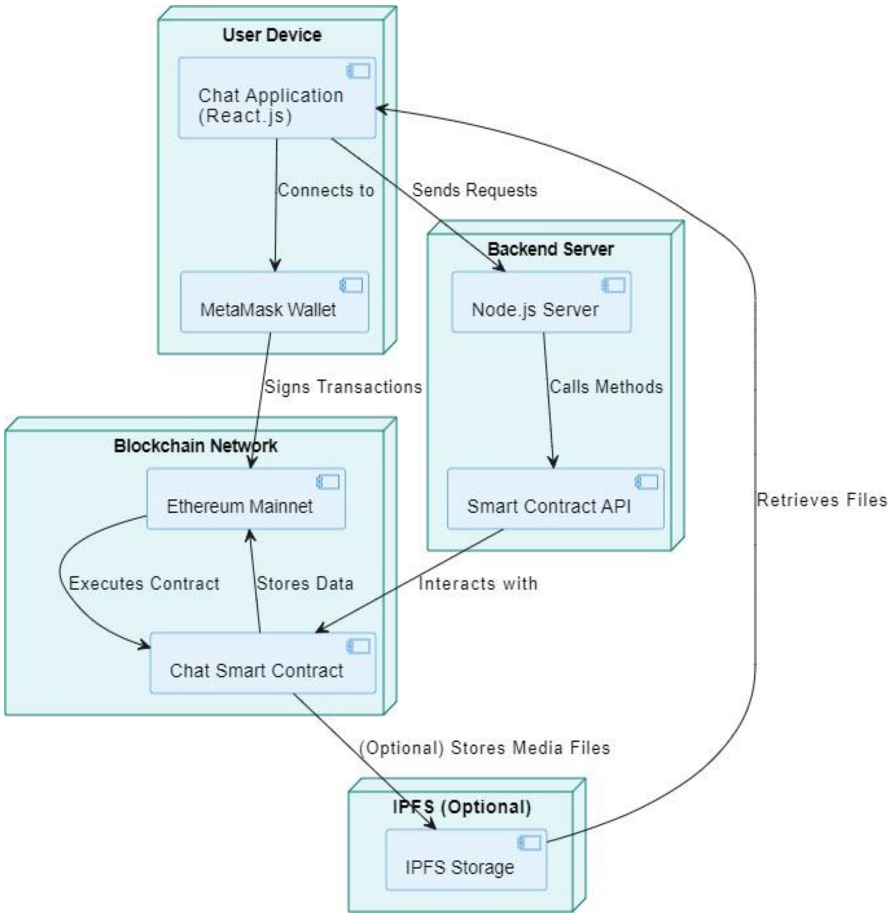


Fig. 1. System Architecture

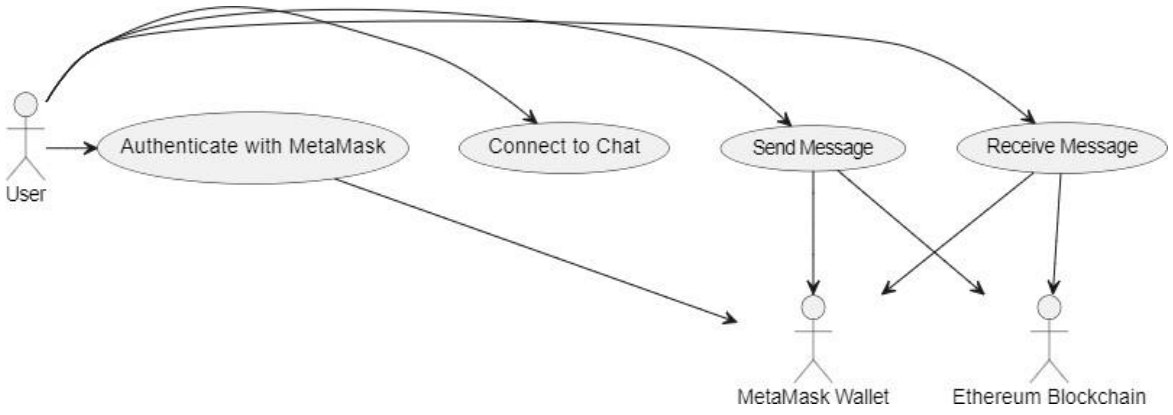


Fig. 2. Use Case Diagram

Fig. 2. illustrates the use case Diagram of our project. The user connects to Chat, can send and receive message, linked through MetaMask and Ethereum Blockchain. Brief discussion on the frontend (user interface), backend (server in- teraction with blockchain), and smart contract on the Ethereum blockchain.

A. Front-End

1. **User Interface (UI/UX):** The user interface is developed using the front-end package js. This actually gives users an interactive and friendly interface whenever they input into this decentralized chat system. Features include MetaMask integration for authentication, an account form to create or even develop a friend’s account, interfaces of send and receive messages in the chat amongst other features.
2. **MetaMask Integration:** MetaMask is an application de- signed to be used both in browser extension form and on mobile devices. That’s how it will authenticate-hand private keys, allowing users access

to the Ethereum blockchain. The interface allows users to sign in through MetaMask; this then links the Ethereum address to the decentralized chat

3. **Web3Modal and web3.js/ethers.js:** All interactions with the blockchain on the frontend would be managed through the use of web3modal and web3.js/ethers.js. The web3.js/ethers.js libraries basically connect the React application to the Ethereum blockchain, allowing features such as fetching data from the blockchain, executing transactions, and interaction with smart contracts.

B. Backend (Node.js)

The sequence diagram from Figure. 3. gives an overview of how the interaction flow of the decentralized chat application takes place, with a spot about function in the backend. As the process is initiated by connecting the MetaMask wallet through the user's React frontend, MetaMask then authenticates and broadcasts the transaction to the Ethereum blockchain for recording in a smart contract. The backend-verified transaction, monitoring the exchange of data between the blockchain and the frontend-using Node.js End. The back- end extracts the messages from the smart contract; afterward, the result goes to the frontend for protection of user access.

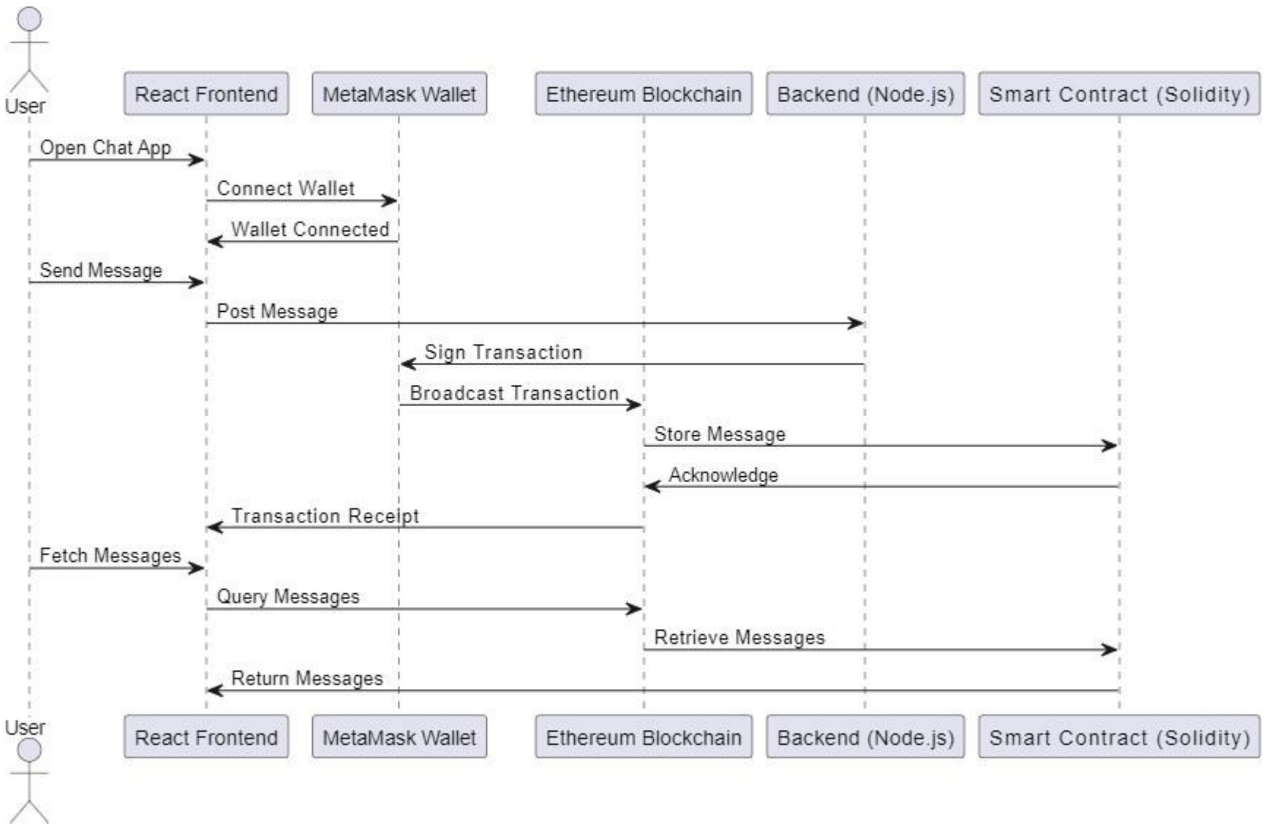


Fig. 3. Sequence Diagram

1. **js Server:** The node server is developed with the help of Node.js that acts as an interface between the frontend and the Ethereum blockchain. Although the application is na- tively decentralized, the auxiliary functionalities like managing friend lists, sessions of users, and accessing the blockchain to process intricate data are taken care of by the Node.js server.
2. **Blockchain Interaction:** With the usage of the two libraries, web3.sol and ethers.js, the backend will interact with the Ethereum network. In simple terms, it will perform transactions—for example, writing (composing) a message to the blockchain—and also retrieve data, such as fetching messages for a specific chat session. These transactions are carried out on the Ethereum mainnet, which means the data is fully decentralized. Each transaction incurs a gas fee, which involves paying a small amount of Ether for processing and confirming transaction details on the network. MetaMask gets utilized pretty heavily for sending transactions and managing them quite effectively nowadays. The

exact gas fee depends on network congestion and the complexity of the transaction. A typical chat message write might cost users 0.001–0.005 ETH through MetaMask, depending heavily on prevailing gas prices at that time.

C. Smart Contracts (Solidity)

Smart Contracts are used to self-executing, automated con- tracts with terms and conditions directly written in the code. The System comprises various sections, discussed below:

1. **Message Management:** Through the implementation of smart contracts using the Solidity program language, they get deployed on the Ethereum blockchain and enable the basic functionalities involving storage, retrieval, and validation of messages being communicated between the users. Since all these messages exchanged between the users are transmitted as a blockchain transaction, they become impossible to change and accessible only through blockchain queries.
2. **Friend Management:** All the above smart contracts would take care of friendship lists and connections among users. Each user would be adding another on the Ethereum addresses, saved on the blockchain, to their account so that keeping privacy and security without a centralised server.
3. **Fees:** Since all messages and interactions-like when adding friends-are recorded on the blockchain, users have to pay transaction fees for each transaction. The smart contracts, in this case, need to be optimized in ways that reduce the gas costs-a very important aspect of the system architecture.

D. Ethereum Blockchain

Ethereum is a decentralized platform used to runn smart contracts. It is a blockchain network which uses proof-of-stake consensus mechanism secure transactions and new blocks. It follows some set of rules called Ethereum protocol allowing peer-to-peer networking, Censorship restraint and an open internet. The following key features makes Ether ideal our research:

1. **Immutable Data Storage:** It is immutable data storage. The Ethereum blockchain is a distributed ledger that hosts all the data linked with chat activities; thus, when messages are sent, it writes them down as transactions and their immutability and verifiability are ensured. This obliterates the need for a central database, thereby making it more secure.
2. **Immutable Data Storage:** The application run on the Ethereum mainnet; this means all transactions will be safe, including the one facilitated by Ethereum's consensus proto- col; besides, its network will afford access to a vast, secure, and reliable blockchain infrastructure.

E. MetaMask Authentication

MetaMask essentially provides a digital wallet for the user and allows safe private keys management during extraction from the application. That would mean that a user can sign transactions and authenticate within the chat application to allow messaging or adding contacts on the user end. By this mechanism, it ensures decentralized identity verification; in other words, no dependence on traditional authentication systems such as usernames and passwords.

DATA FLOW IN THE SYSTEM

As shown in figure 4, the process of building the entire de- centralized chat application based on the Ethereum blockchain and MetaMask for secure messaging involves numerous steps. Initially, a user launches a chat application that connects with his/her MetaMask wallet. On successful wallet connection, the user is thereby authenticated; otherwise, an error message is shown.

Upon logging in successfully, the user composes a mes- sage and sends it to start a transaction signing request via MetaMask. If the signing of the transaction goes well, it will be sent to the Ethereum blockchain, and the message will be safely stored in a smart contract. If, however, it is not signed correctly, a signing error will be shown to the user.

Then the system retrieves the stored messages from the blockchain and gets the history of messages in case of success.

The messages are shown to the user in case of success; if retrieval fails, a retrieval error message will be shown. This workflow allows secure and decentralized storage and retrieval of chat messages using blockchain immutability for storage and the MetaMask trusted transaction interface for backward access.

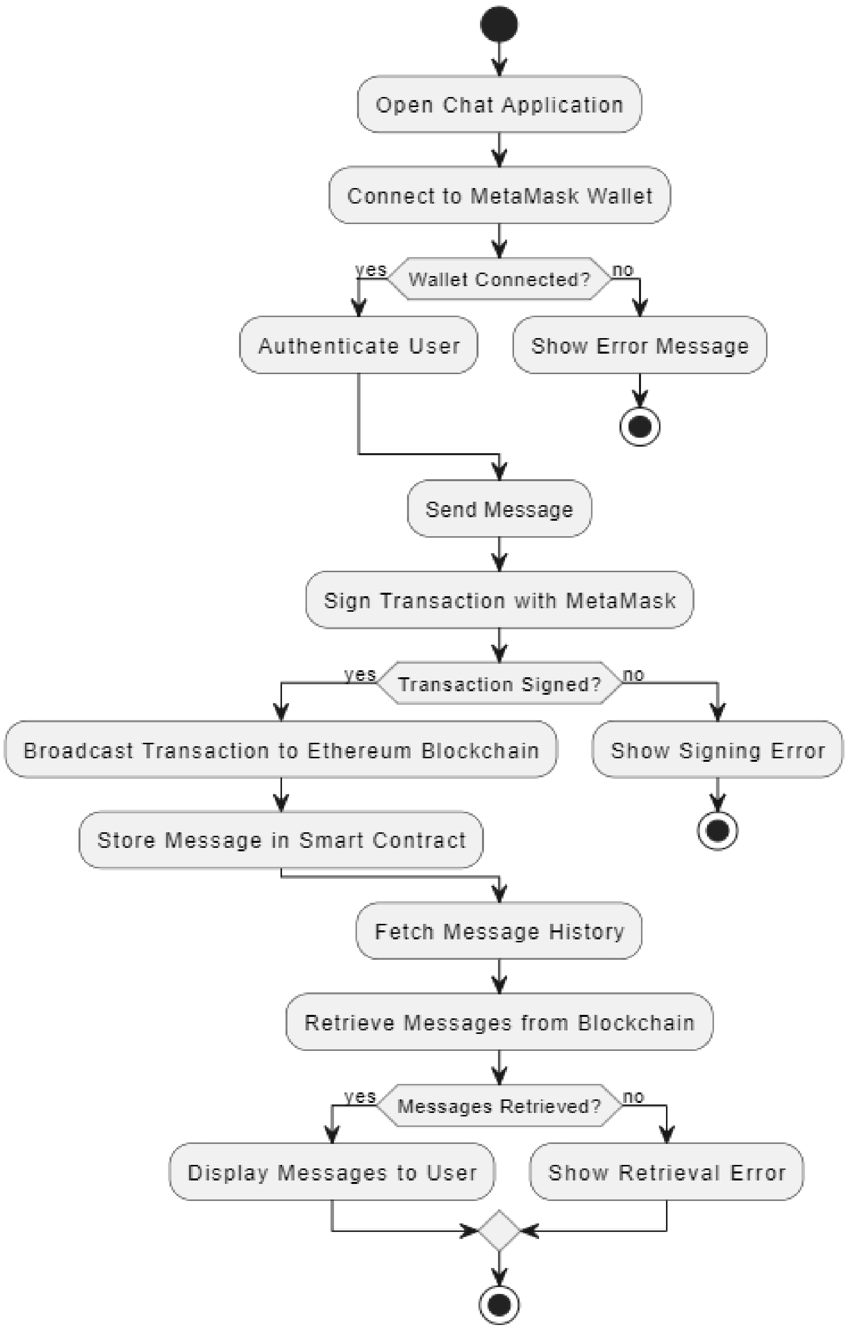


Fig. 4. Flow-Chart of this system

The decentralized chat application employs a secure, blockchain-based communication framework that ensures user privacy and data integrity. Below are the formal steps:

1. **Authentication:** Users are authenticated through Meta- Mask whenever the application starts. An Ethereum address linked at login becomes a user’s identity within the application

2. **Account Creation:** Once authenticated, a user can create an account and link his Ethereum address with his profile.
3. **Adding Friends:** A user can look up or add friends using their Ethereum addresses, verified through a smart contract
4. **Sending Messages:** When a user sends a message, it is encrypted. The encrypted message is sent to the Ethereum blockchain via a smart contract transaction. The message, along with the necessary metadata, is stored as a
5. **Receiving Messages:** The blockchain sends the en- crypted message to the recipient. The recipient decrypts it using his private key, so the message now only remains readable to him.

SECURITY CONSIDERATIONS

Security is a fundamental concern in the development of decentralized applications (dApps), especially in chat appli- cations where user privacy is paramount. The decentralized nature of dApps eliminates the need for a central authority, reducing the risk of data breaches and censorship.

A. Decentralization and Privacy

1. **Decentralization:** Using blockchain technology, the chat system prevents all types of single points of failure found in traditional, centralized systems. It does not have some central server that may be attacked or worse, compromised because all its data-message and information on users-is spread across many nodes of the Ethereum network.
2. **Privacy:** All the messages are stored in a blockchain such that every message is encrypted through cryptography before sending it to a party, such that the sender and recipient are the only parties given keys to access the content of the message, though the transaction is recorded publicly.

B. Authentication and Authorization

1. **MetaMask Authentication:** MetaMask provides a secure system of authentication based on Ethereum private key in that only a transaction can be sent by the owner of a wallet; the dApp need not store or maintain sensitive information like usernames and passwords even more so their private keys.
2. **Non-repudiation:** Since a transaction is linked to an Ethereum address, users cannot formally deny sending a message. Because the blockchain is decentralized, everything- message, friend requests-is verifiable and immutable.

C. End-to-End Encryption

1. **Message Encryption:** Asymmetric encryption were used to encrypt the message before it is written in the blockchain. The message will only be encrypted by the public key of the intended recipient. Only the private key of the intended recipient shall decrypt the message.
2. **Immutability:** One cannot change the data recorded on the blockchain-it is irrevocable. This feature ensures that nobody can modify or alter data in any way and hence points to the message itself in relation to integrity.

D. Smart Contract Security

Smart contracts are at the core of our decentralized chat application and require robust security to prevent critical failures and financial loss. Given the immutable nature of deployed contracts on the blockchain, we employed both automated tools and manual reviews to validate our contracts.

1. Audit and Testing Tools: We used the following industry- standard tools to test and analyze our smart contracts:

- **Slither:** Conducted static analysis to detect vulnerabilities such as reentrancy, uninitialized variables, and redundant
- **MythX:** Performed symbolic execution to identify secu- rity issues like integer overflows, improper authorization, and denial-of-service threats.
- **Hardhat Coverage:** Verified that all functions and code paths were properly tested, ensuring high test coverage.

2. Vulnerability Mitigation Techniques: The following vul- nerabilities were carefully addressed in our implementation:

- **Reentrancy:** Prevented by using the Checks-Effects- Interactions pattern and by restricting external contract calls within state-changing functions.
- **Front-running:** Minimized by designing non- incentivized and transparent transactions that are less sensitive to ordering.
- **Denial-of-Service (DoS):** Loops over dynamic arrays were avoided. Instead, mappings were used for friend and message management to achieve constant-time op-
- **Integer Overflow/Underflow:** Solidity version 8+ was chosen for its built-in overflow protection.
- **Access Control:** All contract functions are secured with MetaMask-based wallet verification, ensuring actions are tied to valid Ethereum addresses.

E. Protection from External Threats

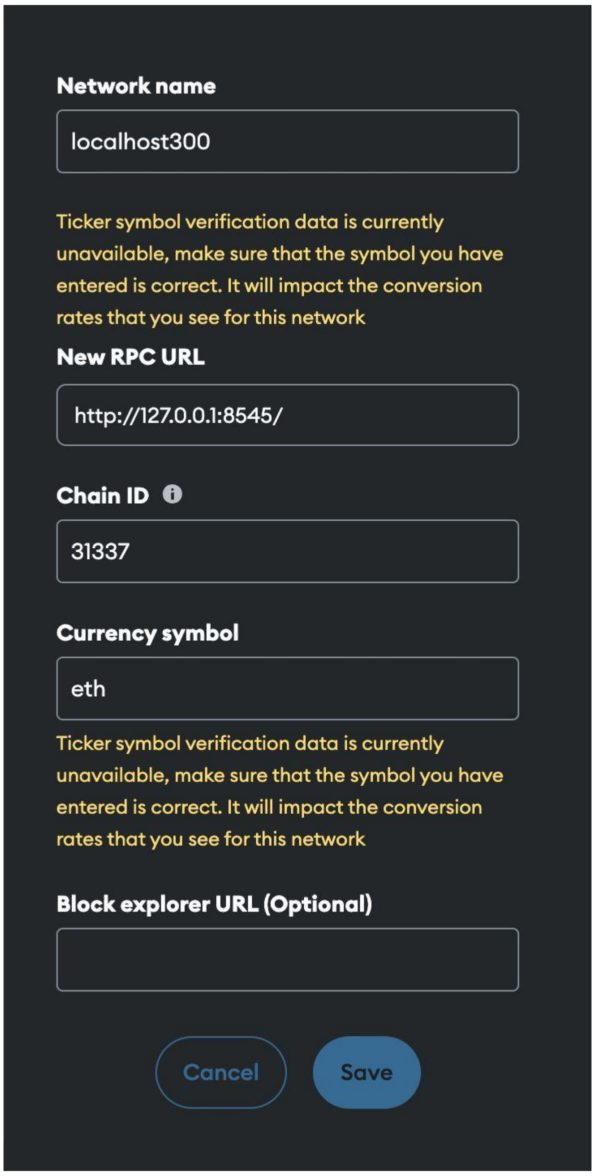
While smart contracts offer strong on-chain security, the surrounding application must also guard against off-chain and network-level threats. Our system includes several mecha- nisms to ensure end-to-end protection.

1. **Phishing and Man-in-the-Middle (MitM) Attacks:** Meta- Mask handles transaction signing locally, preventing the ex- posure of private keys to the browser or the network. The user must manually approve each transaction, making phishing attempts significantly harder. As a result, transaction origin is always traceable and tamper-proof.
2. **Authentication and Identity Protection:** Authentication relies on Ethereum wallet signatures instead of traditional credentials. This method eliminates risks like password reuse or centralized database breaches. Since the keys never leave the user's MetaMask, impersonation is not possible.
3. **Replay and Spoofing Protection:** Ethereum natively en- sures protection against replay attacks using transaction nonces and Additionally, all messages are digitally signed by the sender's private key, making message spoofing infeasi- ble.
4. **Secure Transport Layer:** Interactions between the fron- tend and backend (e.g., for session handling or retrieving meta- data/IPFS hashes) are performed over HTTPS. This ensures that even auxiliary off-chain data remains secure in transit.

RESULT AND DISCUSSION

The decentralized chat application developed in this re- search is successfully implemented on blockchain for secure and private messaging. Here are the visualization of our final product. For testing the functionalities of our contract we have manually created an localhost network in Fig. 5. Above mentioned is the credentials of that network.

Figs. 6-9 depicts the login interface welcomes users where they enter their credentials to **log in** or **sign up**. The interface is modern sleek design, used by blockchain developers, high- lighting social and community focused environment. The Chat Interface is dark-themed with various sections where users can communicate with each other, aligned with our project objective for safe, secure communication channels focusing on AI adaptability. Further integration with an agriculture-focused community, along with additional functionalities, is envisioned to enhance accessibility and support within the agricultural sector.



Network name

localhost300

Ticker symbol verification data is currently unavailable, make sure that the symbol you have entered is correct. It will impact the conversion rates that you see for this network

New RPC URL

http://127.0.0.1:8545/

Chain ID

31337

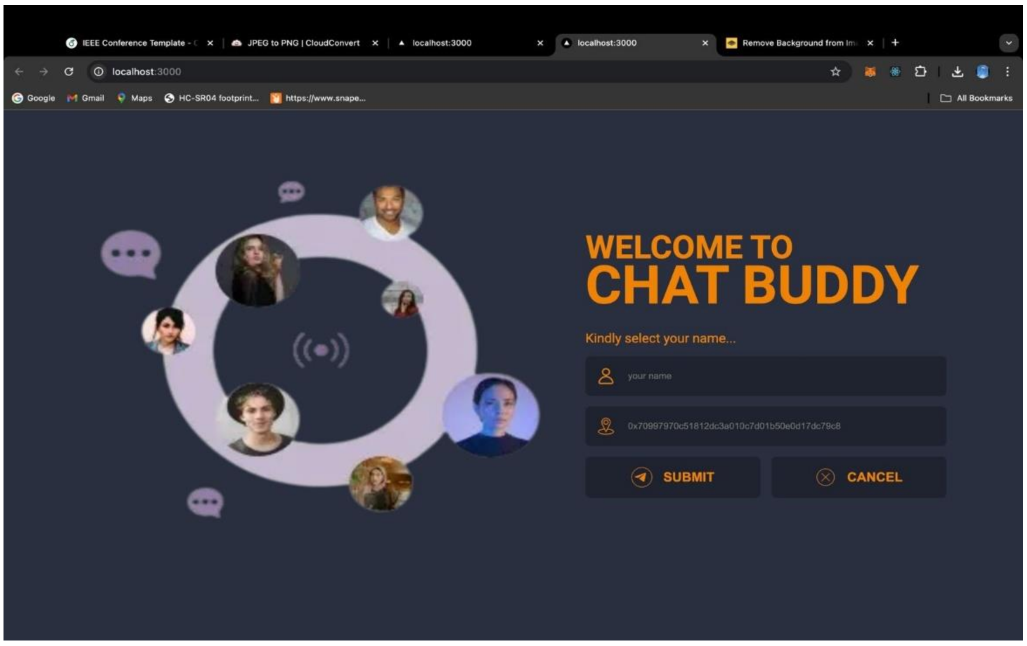
Currency symbol

eth

Ticker symbol verification data is currently unavailable, make sure that the symbol you have entered is correct. It will impact the conversion rates that you see for this network

Block explorer URL (Optional)

Fig. 5. Testing URL with network



WELCOME TO
CHAT BUDDY

Kindly select your name...

your name

0x70997970c51812dc3a010c7d01b50e0d17dc79c8

Fig. 6. Login Interface

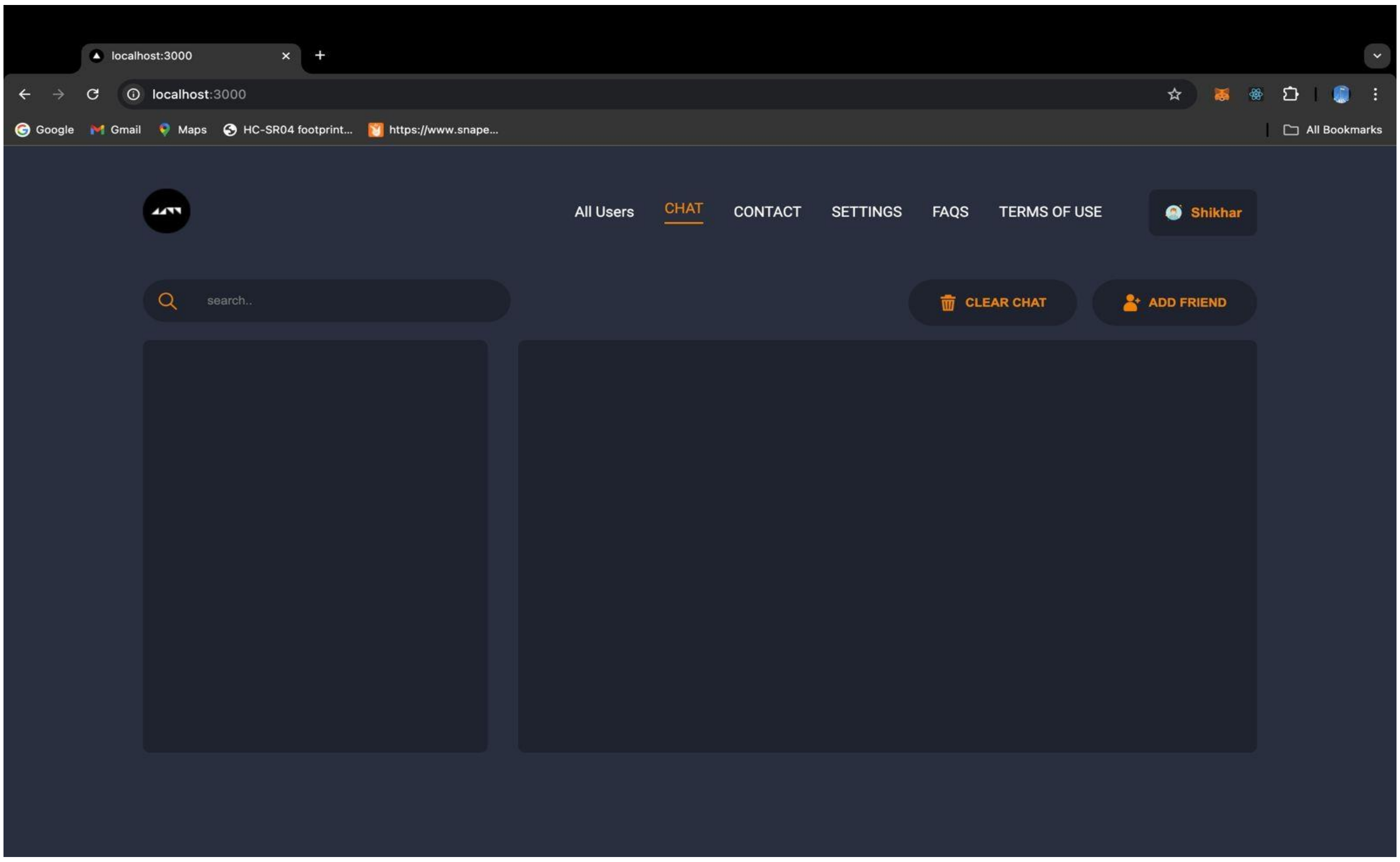


Fig. 7. User Chat

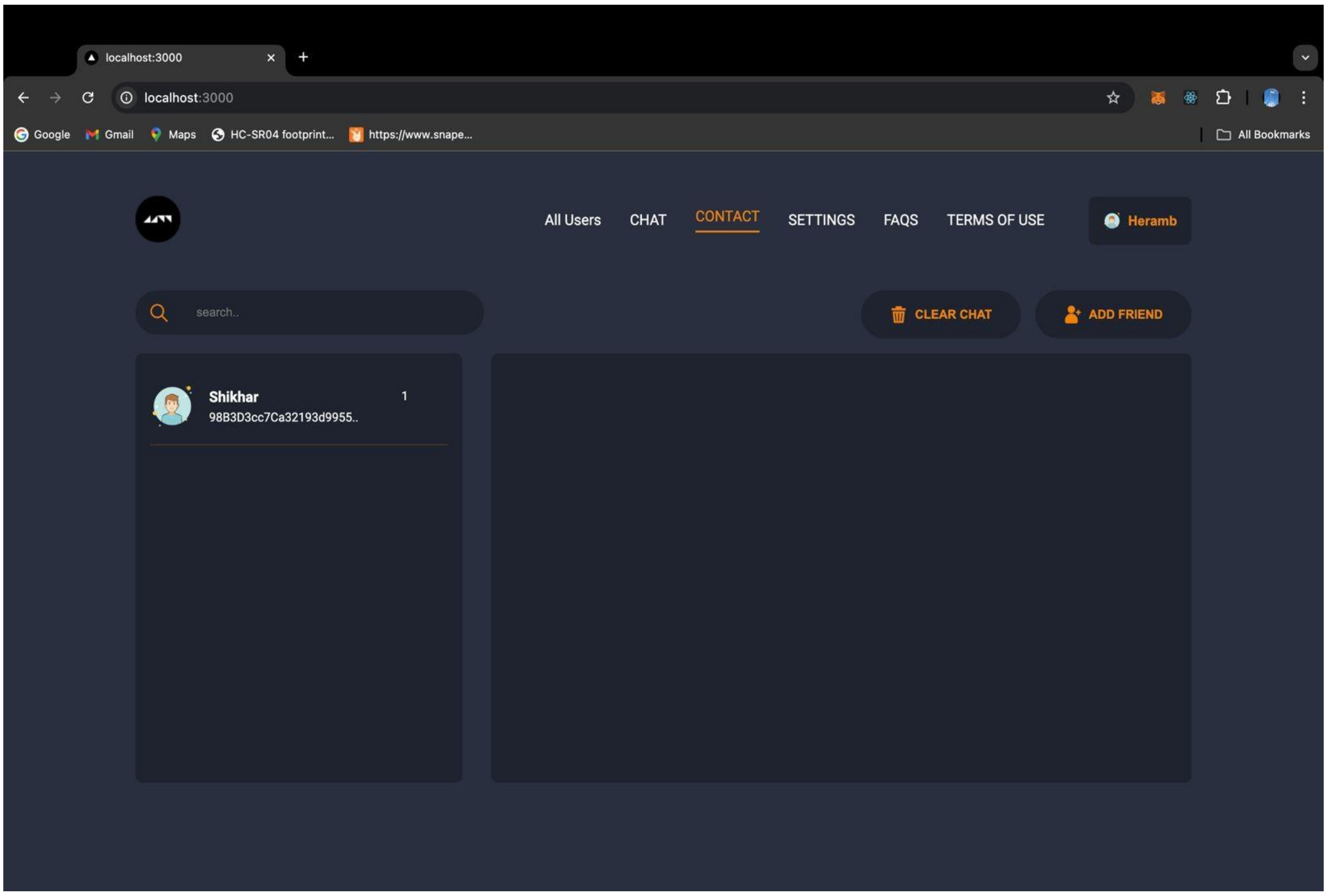
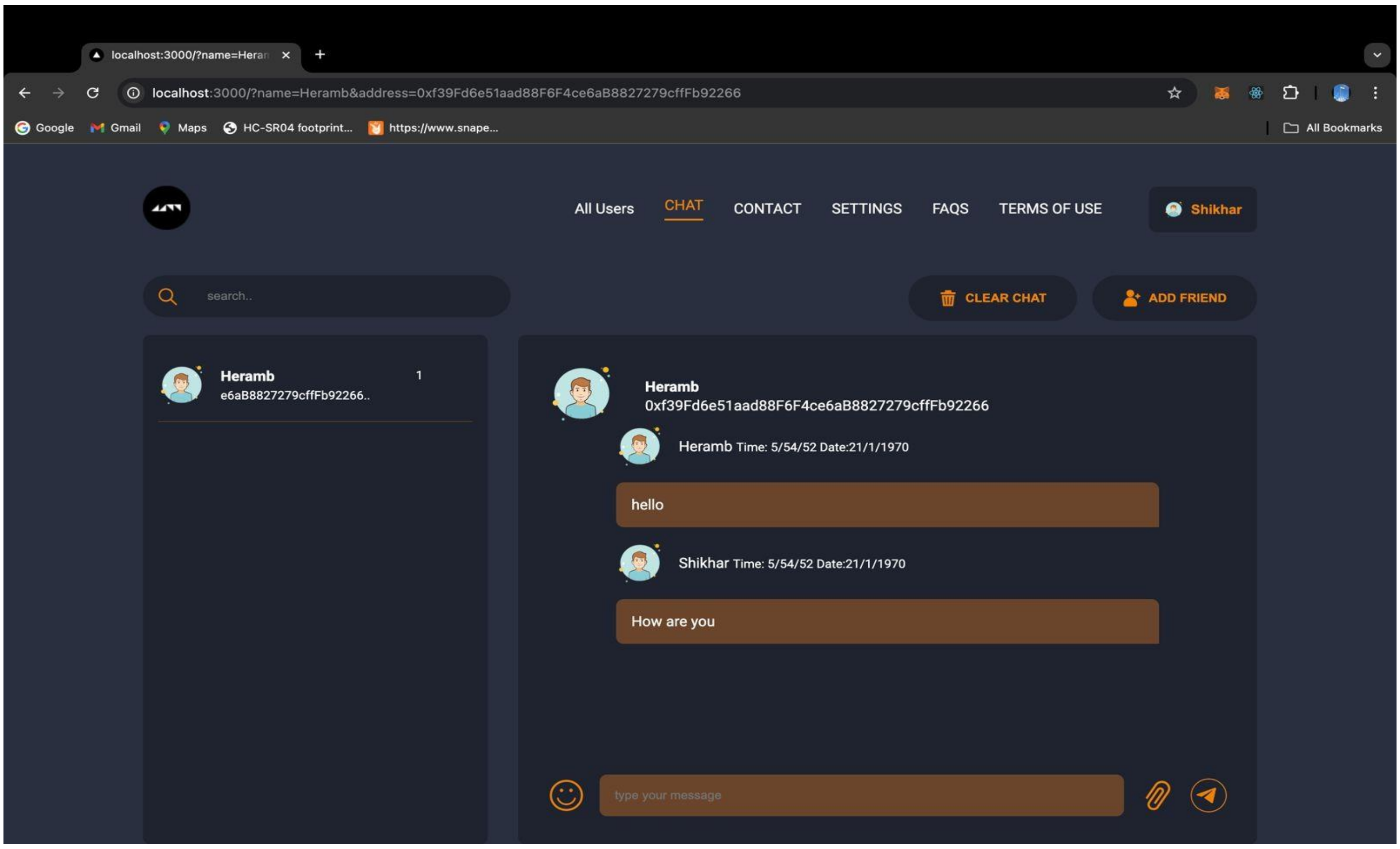


Fig. 8. Card Token



Heramb
e6aB8827279cffFb92266..
1

Heramb
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266

Heramb Time: 5/54/52 Date:21/1/1970

hello

Shikhar Time: 5/54/52 Date:21/1/1970

How are you

Fig. 9. Chat Interface

CONCLUSION AND FUTURE SCOPE

That is how blockchain technology proves its power in inno- vative methods such as online communications by delivering secure decentralization to chat applications. Its use in the ether will ensure some of the above-mentioned properties of data immutability, privacy, anti-censorship, and protection from data compromise that improve user secure interaction across this application. For the immediate future, like in the next year or so, this work promises to yield a very interesting outcome: An AI-based agricultural chatbot is working as an intelligent assistant for farmers. The system will integrate real-time crop health monitoring by IoT sensor integration for timely detection of abnormalities based on soil moisture, pH values, temperature, and other environmental data. Predictive analytics through machine learning models—for example, Long Short- Term Memory (LSTM) networks—and Random Forest clas- sifiers for pest and disease predictions will be employed to provide actionable insights for the farmer's productivity. The chatbot will feature multilingual natural language processing to enable access to speaking and usage by varied linguistic groups. Provided with secure and tamper-proof recording of crop data, pesticide utilization, and transactions in a supply chain, because transparency and traceability will thus be in- creased. The final piece of this system will be the user-friendly web and mobile interface integrated into MetaMask for direct interaction with the blockchain. Their intent has primarily been the conversion of the chatbot to an all-purpose digital farming assistant that would enhance agricultural productivity while promoting sustainable, data-centric agricultural practices.

REFERENCES

1. Salfin, S., Kurniadi, P., & Erwin, E. (2024). Language Development in the Digital Age, A Literature Review on the Influence of Technology on Human Communication. Sciences du Nord Humanities and Social Sciences, 1(01), 01-07
2. Alencar, F. C., Ferreira, C. H., & Filho, D. L. (2024, May). Developing Decentralized Applications: A Framework Approach on Blockchain Net- In Proceedings of the 20th Brazilian Symposium on Information Systems (pp. 1-10).

3. Mars, M., & Scott, R. E. (2016). WhatsApp in clinical practice: a literature review. The Promise of New Technologies in an Age of New Health Challenges, 82-90.
4. Bogos, C. E., Mocanu, R., & Simion, E. (2023). A security analysis comparison between Signal, WhatsApp and Cryptology ePrint Archive.
5. Stephen, R., & Alex, A. (2018, August). A review on blockchain In IOP conference series: materials science and engineering (Vol. 396, No. 1, p. 012030). IOP Publishing.
6. Buterin, V. (2016). Ethereum: platform review. Opportunities and chal- lenges for private and consortium blockchains, 45, 1-45.
7. Wohrer, M., & Zdun, U. (2018, March). Smart contracts: security patterns in the ethereum ecosystem and solidity. In 2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE) (pp. 2-8). IEEE.
8. Lee, (2023). Using the MetaMask Crypto-Wallet. In: Begin- ning Ethereum Smart Contracts Programming. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-9271-6-5
9. Chen, X., He, S., Sun, L., Zheng, Y., & Wu, C. Q. (2024). A Survey of Consortium Blockchain and Its Applications. Cryptography, 8(2), 12.
10. Ellewala, P., Amarasena, W. D. H. U., Lakmali, H. S., Senanayaka, L. M. K., & Senarathne, A. N. (2020, December). Secure messaging platform based on blockchain. In 2020 2nd International Conference on Advancements in Computing (ICAC) (Vol. 1, pp. 317-322). IEEE.
11. Yi, H. (2019). Securing instant messaging based on blockchain with machine learning. Safety Science, 120, 6-13.
12. Ranganthan, P., Dantu, R., Paul, A., Mears, P., & Morozov, K. (2018, October). A decentralized marketplace application on the ethereum blockchain. In 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC) (pp. 90-97). IEEE.
13. Liu, H. (2024). Designing And Implementing a Chat System with Enhanced Security Via AES Encryption Highlights in Science, Engineering and Technology, 85, 480-486.
14. Venkatesan, K., & Rahayu, S. B. (2024). Blockchain security enhance- ment: an approach towards hybrid consensus algorithms and machine learning techniques. Scientific Reports, 14(1), 1149.
15. Wan, J., Li, J., Imran, M., & Li, D. (2019). A blockchain-based solution for enhancing security and privacy in smart factory. IEEE Transactions on Industrial Informatics, 15(6), 3652-3660.s