International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 11 | Nov - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Blocklance - A Blockchain Powered Freelancing Platform

KHEDER MOHAMED ANEESULLA[1], KISHAN KUMAR MISHRA[2], KSHITIZ MOWAR[3], MAULIK SHARMA[4] and Dr. SMITHA PATIL[5]

[1,2,3,4]*Department of Computer Science and Engineering, Sir M. Visvesvaraya Institute of Technology, Bengaluru, Karnataka, India*

[5]*Assistant Professor B.E., M.Tech, Ph.D., Department of Computer Science and Engineering, Sir M. Visvesvaraya Institute of Technology, Bengaluru, Karnataka, India*

Abstract: The global gig economy relies heavily on centralized freelance platforms, which face many problems like high fees for intermediaries, payment disputes, and a lack of transparency. Clients may not receive their work after paying, and freelancers can go unpaid after delivering their work. This paper presents BlockLance, a complete decentralized application (dApp) aimed at solving these problems using blockchain technology. At its core, the system uses a Solidity smart contract deployed on an EVM-compatible testnet (Sepolia) that acts as a trustless, automated escrow agent. The contract oversees the entire job process, from job creation and funding tied to milestones to automatic payment release when the client approves the work. This on-chain logic is managed via a React-based frontend that connects to a Node.js/Express backend for off-chain tasks, such as file uploads and user data. The setup ensures that all financial transactions are safe, transparent, and verifiable on the blockchain, while user-related data is processed effectively off-chain. The successful launch and implementation of BlockLance show a practical model for a more fair and safer freelance environment, effectively reducing the probability of payment fraud and dependence on expensive central authorities.

Index Terms— Blockchain, Decentralized Application (dApp), Escrow, Gig Economy, Smart Contracts, Solidity.

I. INTRODUCTION

The freelance, or "gig" economy has grown explosively to become a substantial element of the world's workforce, transforming many of the paradigms of traditional employment. Digital platforms like Upwork, Fiverr, and Toptal have served as accelerants, establishing huge markets that link millions of customers with freelancers worldwide [1]. These systems create real value in terms of access and market exposure. Their centralized design has led to a myriad of structural ills.

These platforms act as trusted third parties, maintaining total and non-transparent control over user data, financial transactions, and most crucially-dispute resolution [2]. Herein lies a number of points of friction:

Exorbitant Fees: Commission fees can range from 10% to 20% or more, usually charged by platforms, which highly extract significant value from both clients and freelancers.

Payment Delays and Disputes: Freelancers often have to wait for a long period of time for the clearance of their payments, while in case of disputes, decisions are given by a central authority, which can be quite arbitrary and absolute, thereby being very unfair to the concerned party.

Lack of trust is at the core of the problem: a mechanism of exchanging value in a trustless way. A client doesn't want to pay in advance for work he hasn't received yet, and a freelancer doesn't want to deliver work without receiving any payment for it.

Traditional platforms try to solve this "trust problem" by acting as a centralized escrow agent. However, this service is the primary justification for their high fees, and is often slow, cumbersome, and subject to the platform's own biases [3].

Such a novel and powerful solution comes with blockchain technology, and more concretely, smart contracts. A smart contract can be used as a fully automated, transparent, immutable escrow agent that performs its logic programmatically without the intervention of any human intermediary [4].

This paper covers the design, implementation, and deployment of BlockLance, a proof-of-concept decentralized freelance platform. In the BlockLance platform, the logic of a transparent, automated, and trust less escrow system is realized through a Solidity smart contract on the Sepolia testnet. It achieves this by locking client funds into the contract at the start of a job and automatically releasing them to the freelancer only upon the completion and client-side approval of verifiable milestones, thereby dispensing with any need for a trusted intermediary in the process.

The key value of this research is the elaboration of a functional full-stack hybrid architecture. We demonstrate how to effectively combine on-chain logic for finance and state with an off-chain backend for large file storage and metadata and a modern web frontend. This paper describes the architecture of the system, its implementation, and successful test-case

evaluation, proving a practical, secure alternative to existing centralized freelance marketplaces.

II. LITREATURE SURVEY

The challenges of centralized freelance platforms have been well-documented. Chakraborty et al. [2] provide a comprehensive analysis of the "gig economy ecosystem," identifying the high commission fees and the centralized, often opaque, dispute resolution process as major points of friction that create a precarious environment for freelancers. Their work lays the theoretical groundwork for a "secured and transparent freelancing ecosystem" based on blockchain.

Several blockchain-based solutions have been proposed to address these shortcomings. Early models focused mainly on the use of cryptocurrency for direct peer-to-peer payments [5]. While this avoided platform fees, it didn't resolve the fundamental escrow problem; it just moved all the trust risk onto one of the two parties.

More sophisticated suggestions then emerged which implemented smart contracts to govern the freelancer contract. Rehman et al. [3] presented a "secure and transparent freelancing platform" based on a smart contract that can hold funds. The model they put forth does validate the concept, but it is heavily focused on on-chain logic without explaining a full-stack implementation of how large deliverables (file transfers) would be handled.

The work of Verma et al. [4] was one of the key literature pieces that studied in detail a "decentralized escrow system" using smart contracts. Their work essentially proves that a smart contract can act as an automated agent to hold and release funds based on programmatically set rules, which forms the basis for the BlockLance escrow mechanism.

In addition to these, there is the important notion of on-chain reputation. Liu et al. [6] have proposed a "blockchain-based decentralized reputation system" for online marketplaces. Though BlockLance does not currently implement a reputation system, this body of work identifies a crucial next step because reputation provides a metric for trust in the absence of a central vetting authority.

BlockLance builds directly upon this body of work. It takes the theoretical frameworks of Chakraborty et al. [2] and Rehman et al. [3] and the proven escrow logic from Verma et al. [4] and integrates them into a functional, end-to-end, milestone-based system. Its novelty lies in integrating this on-chain logic with a modern, full-stack web application, including an essential off-chain component for handling large file (deliverable) uploads, therefore presenting a complete and practical system rather than a purely theoretical model.

III. METHODOLOGY

The BlockLance platform is built as a complete, three-tier decentralized application. Its architecture is carefully crafted to harmonize the blockchain's security, immutability, and transparency with the performance, cost, and storage requirements of a contemporary web application.

A. System Overview

The system is composed of three primary components:

1. *Blockchain Layer (On-Chain):* A Solidity smart contract, FreelanceMarketplace.sol, resides on the Sepolia EVM-compatible testnet. This layer manages all financial transactions, job state, and escrow functions.
2. *Backend Layer (Off-Chain):* A Node.js/Express server, hosted on Render, takes care of tasks that aren't ideal for the blockchain, including large file uploads and the storage of non-essential metadata.
3. *Frontend Layer (Client-Side):* The user interface is implemented as a React (Vite) application, hosted on Netlify. This layer facilitates interaction with blockchain, using the user's wallet, and the backend server.

B. Blockchain Layer (On-Chain):

The FreelanceMarketplace.sol smart contract contains the system's fundamental logic. This contract serves as the definitive authority for all transactions involving value.

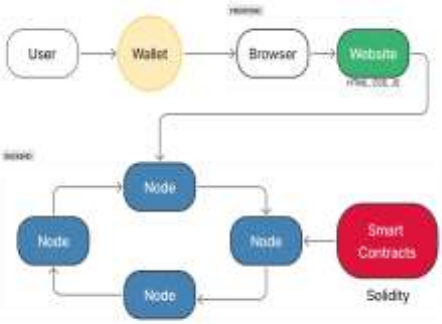
1. *State Variables:* The contract employs data structures to govern the platform's state. These include a Job struct, which encompasses jobId, clientId, freelancerId, title, description, budget, and state, among other attributes, and a Milestone struct, which contains a description, amount, state, and deliverableUrl. Furthermore, mappings, specifically mapping(uint256 => Job), are utilized for the storage and retrieval of job data.
2. *Core Functions:* The core functions of the application are createJob(), which fixes the client's funds in escrow; acceptProposal(), which assigns a freelancer to the job; submitMilestone(), which points to off-chain functions ; and approveMilestone(), which automatically pays the freelancer from the escrow.
3. *Security Considerations:* Access control is implemented using modifiers (e.g., only Client, only Freelancer) as well as require() statements in the code.

C. Backend Layer (Off-Chain)

A lightweight Node.js/Express server provides appropriate off-chain services, mainly for dealing with large file uploads (multipart/form-data via multer) that cannot be stored on-chain. The server saves deliverables and returns a static URL which is stored in the smart contract as a string.

D. Frontend Layer (Client-Side)

A React application utilizes the Ethers.js library [7] in order to connect to the user's MetaMask wallet. Thus the user seamlessly confirms transaction approvals and interacts with the smart contract. The frontend serves as an example of a hybrid dApp model by first communicating with the off-chain backend to upload the file and, after a successful upload, calling an on-chain transaction to register the file upload URL.



IV. RESULTS

The complete BlockLance application was successfully developed and deployed across the web-3 architecture. The system's effectiveness was validated through functional testing of the entire job lifecycle and analysis of its performance.

A. Smart Contract Deployment and Verification:

The smart contract is compiled using Hardhat and deployed to the Sepolia testnet. The deployment process was successful, and the contract's source code is verified on Ether-scan. This result confirms that the contract is publicly auditable, and its functions can be interacted with as designed. All on-chain state changes (e.g., job creation, payment) are visible and verifiable through a blockchain explorer, hence enhancing transparency.

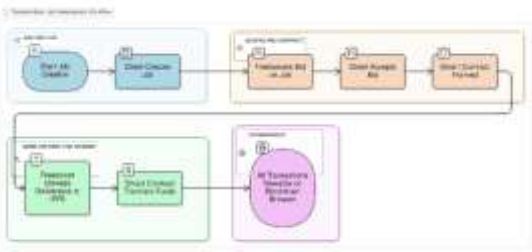
B. Use Case Scenario for Full Job Lifecycle:

A comprehensive test conducted to simulate a real-world client-freelancer interaction with a mock job:

1. *Job Creation:* A Client account (Account A) successfully created a new job with a 0.1 sepETH budget split into 2 milestones. The 0.1 sepETH was transferred from Account A and locked in the smart contract's escrow, as confirmed to be present on the Sepolia Etherscan.
2. *Proposal and Acceptance:* A Freelancer account (Account B) submitted a proposal, which Account A accepted. The job's state on-chain was updated from Open to InProgress, and Account B was locked as the freelancer Id for that job.
3. *Milestone 1:* Account B uploaded the deliverable via the frontend to IPFS. This file is received by the IPFS server. Account B then submitted the file's URL, which is recorded in the smart contract.
4. *Payment 1 (Automated):* Account A approved the milestone. The smart contract's internal logic was executed as designed, automatically transferring 0.05 ETH from its escrow balance to Account B. This transaction was instantaneous upon block confirmation by miners after transaction is mined, and reflected on Etherscan.
5. *Completion:* The process was repeated for the second milestone, resulting in the final 0.05 ETH payment. The job's final state is correctly set to Complete. This test confirms that the core escrow mechanism is fully functional.

C. System Performance

- Transaction Speed: The average transaction confirmation time ranged from 7 to 26 seconds and was influenced by Sepolia testnet congestion. This shows that L2 scaling is necessary for a production application, but it is acceptable for a testing environment. Backend Response Time: File uploads and API requests were handled by the off-chain backend (running on a free-tier render server) in less than 800 ms, which was adequate for a responsive user experience. Gas Fees: The Sepolia testnet was used to track gas prices. Because of state changes and value transfer, the createJob and approveMilestone transaction functions used the most gas. The design decision to reduce on-chain storage is validated by the analysis of gas consumption, which shows that mainnet deployment would be costly even though it is nominal on a testnet..



V. CHALLENGES FACED

The development of a hybrid decentralized Application presented several unique challenges not found in traditional web2 web application development.

A. On-Chain vs. Off-Chain Data Synchronization: A primary challenge was deciding what data to store on-chain versus off-chain. Storing all data (like job descriptions, proposals, and user profiles) on-chain is expensive and slow. However, storing it off-chain contradicts our decentralized approach. Our team

chose a hybrid model, but this introduced a new challenge, the state synchronization. For example, when a freelancer uploads a piece of work, the frontend must first wait for the backend to confirm the file upload and then wait for the submitMilestone transaction on blockchain transaction to be mined. Building a user interface that gracefully handles these two different, in an asynchronous manner, and variable-length waiting periods was a significant frontend hurdle.

B. User Experience for Web3: Traditional web2 users are not familiar with browser wallets, gas fees, or transaction signing. The largest challenge was abstracting the complexity for normal users. The team had to build a UI that not only looked like a clean Web2 platform but also clearly guided the user through Web3 specific items. This included writing clear instructions for "signing" a transaction with their wallet and providing feedback while a transaction is "pending" (which can take much longer than an HTTP request).

C. Gas Fee Volatility and L1 Viability: High gas fees presented a core design challenge: how to build a system that is economically viable. It became clear that the storage large files or complex data on-chain was impossible. This challenge directly aimed to the architectural decision to use a centralized backend for files using the Piniata service. It also confirmed that while the logic is sound, the application, in its current form, is only economically feasible on a low-cost L2 scaling solution rather than Ethereum mainnet.

VI. CONCLUSION

This paper has proposed BlockLance, a full stack, completely decentralized freelance platform that effectively solves the critical weaknesses of high commission fees and ambiguous payments of the traditional gig economy. The system has been shown to eliminate the need for a central intermediary for financial transactions by leveraging a Solidity smart contract as a trustless, automated escrow agent. The successful end-to-end test case featured in the results proves that the automated, milestone-based payment logic is sound and workable. The hybrid architecture-a safe, on-chain ledger for value and an efficient, off-chain server for data-offers a workable and extensible blueprint for real-world dApps. BlockLance is a strong proof-of-concept that a more secure, transparent, fair future of the gig economy is achievable via decentralized technology.

VII. FUTURE IMPROVEMENTS

While BlockLance provides a good foundation, several aspects can be further researched and developed to create a production ready application.

Identified Limitations:

1. *Centralized File Storage:* The use of a Node js backend system for file uploads creates a single point of failure and some chance of censorship, which runs counter to the decentralized ethos.
2. *Basic Dispute Resolution:* The current model is binary. The client either approves payment or holds it back. It does not account for complex, good-faith disputes over the quality of work, and other subjective disputes.
3. *Public Data:* All job details (title, budget) are stored publicly on the blockchain, which may not be desirable for corporate clients seeking confidentiality. This poses a privacy concern.

B) Avenues for Future Research Future work will focus on addressing these limitations:

1) *Decentralized Storage:* The off-chain backend can be recoded with a decentralized storage system like the Inter Planetary File System (IPFS). Files could be uploaded to IPFS, and only the resulting content-addressed hash would be stored on-chain, fully decentralizing the data layer, without any services.

2) *Decentralized Arbitration:* A dispute resolution system could be added. This could be a third party arbitration service (e.g., Kleros) that uses crypto-economic incentives for a decentralized collection of jurors to review evidence and resolve disputes.

3) *On-Chain Reputation:* An on-chain reputation system could be built. When a job completed, both parties could mint a non transferable NFT (otherwise called a "Soul-Bound Token") or to update a reputation score, building a verifiable, on-chain work history that builds trust without a central authority maintaining control over system.

4) *Privacy Enhancements:* Implementing the use of zero-knowledge proofs (zk-SNARKs) could allow for job details to be verified thoroughly and executed by the smart contract without being publicly revealed on the blockchain to enhance privacy.

VIII. REFERENCES

[1] Irawan Afrianto, Christover Ramanda Moa, Sufa Atin, Iding Rosyidin, and Suryani,

"Prototype Blockchain Based Smart Contract For Freelance Marketplace System," VTU Consortium, IEEE (2021)

[2] Milan Radosavljevic, Aleksandar Pesic, Nenad Petrovic, and Milorad Tosic,
"Freelancing blockchain: A practical case-study of trust-driven applications development," Research gate (2021).

[3] Mihir Gandhi, Priyam Shah, Devansh Solanki, and Mihir Shah,

"Decentralized Freelancing System - Trust and Transparency," IRJET, 2019

[4] S. Chakraborty, S. Aich, H.-C. Kim,
"Blockchain for Gig Economy: Smart Contract Based Secured and Transparent Freelancing Ecosystem," *2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, IEEE, 2019.

[5] M. A. Rehman, M. S. Obaidat, M. K. Shahzad,
"A Secure and Transparent Freelancing Platform using Blockchain Technology," *2021 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)*, IEEE, 2021

[6] H. Cui, R. Wang, Y. Wang, and Z. Zhang,

"A Secure Storage Scheme Based on Blockchain and IPFS for Medical Information," *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, IEEE, 2020.