

Blockchain-Based Smart Contract System for Cross-Border Supply Chain Payments: Reducing Transaction Costs and Settlement Risks for SMEs

Zhou Tang

*School of Business, University College Dublin (Kaplan Higher Education Academy), Singapore,
Singapore
2970318572@qq.com*

Abstract. This study examines whether Ethereum-based smart contracts can help reduce transaction costs and settlement risks in cross-border trade for small and medium-sized enterprises (SMEs). To investigate this issue, a hybrid decentralised application was developed and deployed on the Ethereum Sepolia test network. The system uses a Solidity-based escrow mechanism that locks buyer funds when an order is created and automatically releases payment after delivery confirmation. By removing the need for multiple intermediary banks, the system aims to improve settlement efficiency and transparency. During testing, transaction confirmation was completed in about fifteen seconds, with users only paying blockchain gas fees instead of traditional intermediary charges. Compared with the conventional international wire transfer process, the blockchain-based approach showed clear advantages in settlement speed, transaction cost, and payment visibility. However, several practical limitations were also identified, including cryptocurrency price volatility, fluctuating network fees during congestion, and the difficulty of wallet management for non-technical users. Future research could focus on stablecoin integration and Layer 2 solutions to improve practicality for SME adoption.

Keywords: Blockchain, Smart Contract, Cross-Border Payment, Supply Chain Finance, Small and Medium-sized Enterprises

1. Introduction

Cross-border payments play an important role in international trade, but the traditional correspondent banking system often creates difficulties for small and medium-sized enterprises (SMEs). According to World Bank data, average remittance fees remain around 6.2%, while international transfers may take between one and five business days to complete because transactions pass through several intermediary banks [1]. For SMEs with limited cash flow, these delays and additional costs can create settlement risks and reduce business efficiency.

Traditional cross-border payment systems are commonly associated with three major issues: high intermediary costs, slow settlement speed, and limited transparency [2, 3]. Previous studies suggest that blockchain technology may improve payment efficiency and reduce reliance on intermediaries

[2, 3]. However, many existing studies are based mainly on simulations or theoretical analysis and do not provide fully deployed systems designed for SME use cases [4].

To explore this issue further, this study developed and tested an Ethereum-based decentralised application that simulates the payment process in cross-border trade. The research evaluates whether smart contract settlement can help SMEs reduce transaction costs and minimise settlement-related risks. This study contributes to existing research on blockchain applications in trade finance by presenting results from a fully deployed system rather than a simulation-based model. It also provides practical observations on the opportunities and challenges of using blockchain technology in SME cross-border payments.

2. Related work

2.1. Blockchain in cross-border payment: cost and efficiency evidence

A growing body of empirical research confirms that blockchain technology can improve cross-border payment efficiency. Prior studies demonstrate that smart contract integration can reduce intermediary participation and significantly shorten settlement time from several days to 45 seconds in controlled environments [2]. Empirical evidence further confirms that blockchain technology can significantly reduce payment risks and transaction costs in cross-border trade by eliminating intermediary dependencies [3]. These findings establish a strong empirical basis for the hypothesis that smart contracts can replace intermediary institutions in cross-border payment workflows.

2.2. Enterprise blockchain deployments: real-world evidence

Several empirical investigations have examined the deployment of blockchain-based settlement systems by financial institutions. A recent study examining a major bank's deployment of a private blockchain reported a 50–70% reduction in per-transaction cost and a 70% reduction in settlement time compared to the traditional wire transfer baseline [5]. These cases demonstrate that blockchain-based settlement has been operationalized at scale by major financial institutions, providing a proof of concept that this research extends to the small enterprise context.

2.3. Smart contract implementation for trade finance

Several studies address the implementation of smart contracts for trade settlement. A study on Ethereum-based supply chain management proposed a Solidity contract architecture featuring order creation, delivery confirmation, and automated fund release functions [6]. A complementary study confirmed that hybrid decentralised applications can achieve functional parity with centralised trade finance platforms [7].

A recent simulation study by Quan et al. proposed a Tri-Chain Collaboration framework and demonstrated that such systems could reduce processing times by up to 53.8%, lower transaction costs by 30%, and reduce risk events by 35% [4]. However, this work is primarily based on simulation data rather than real-world deployment and does not address operational constraints faced by SMEs in public blockchain networks. This study directly addresses this gap by providing a fully implemented, cloud-deployed Ethereum application targeting the cross-border trade use case.

3. System design overview

The system is designed to address key inefficiencies in traditional cross-border payment systems for SMEs, particularly in terms of settlement delay, intermediary dependence, and limited transaction transparency. First, the system aims to reduce reliance on intermediary institutions by replacing correspondent bank processes with a self-executing smart contract, reducing per-transaction fees from 2–5% of transfer value to network gas costs only. Second, it seeks to automate settlement by encoding payment release conditions directly into immutable on-chain logic, removing the need for manual processing and reducing settlement time from 3–5 business days to near-instant finality. Third, it enhances transparency and auditability by recording every transaction event on a public, tamper-proof blockchain ledger accessible to both buyer and seller in real time.

The system architecture is guided by four principles: (1) Decentralisation: payment execution is governed by smart contracts without centralized control over fund custody. (2) Automation: Payment release is triggered automatically when predefined contract conditions are satisfied. (3) Transparency: all transaction states are publicly accessible via the Ethereum blockchain explorer (Etherscan). (4) Security: Buyer funds are locked in escrow within the smart contract and can only be released under valid conditions defined in the contract logic.

The system involves four principal participants: (1) Buyer: initiates a transaction and deposits funds into the smart contract escrow. (2) Seller: receives payment upon fulfillment confirmation. (3) Smart Contract: Executes escrow logic and enforces settlement rules. (4) Blockchain Network: provides distributed consensus, immutability, and transaction validation.

4. System architecture

4.1. Overall architecture

The proposed system is implemented as a blockchain-based cross-border payment application deployed on the Ethereum Sepolia test network. The system adopts a three-layer architecture, as illustrated in Figure 1. The architecture consists of:

- (1) Smart Contract Layer: Implements escrow and settlement logic on-chain.
- (2) Application Layer (Frontend): web-based interface enabling users to create and monitor transactions.
- (3) Wallet integration layer: MetaMask-based authentication and transaction signing mechanism.

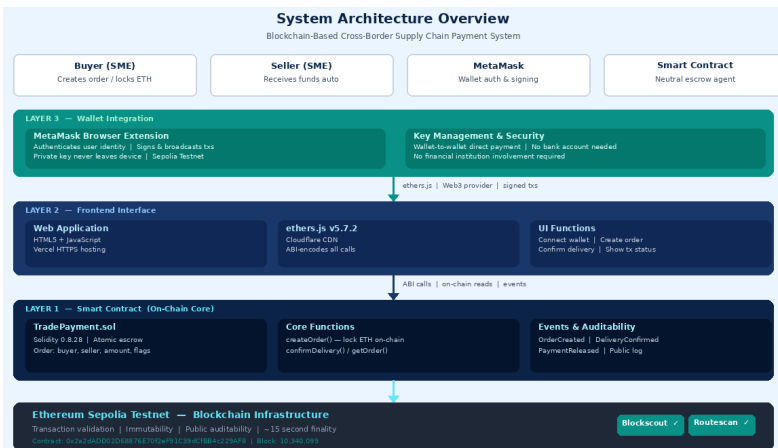


Figure 1. System architecture overview

The smart contract layer forms the core of the system and is responsible for enforcing all payment rules. The frontend layer interacts with the blockchain via the ethers.js library, while the wallet layer enables secure user identity verification and transaction authorization.

4.2. Smart contract design

The core logic of the system is implemented in a Solidity smart contract (version ^0.8.28). The TradePayment smart contract implements a three-phase escrow mechanism that mirrors real-world trade workflows. The contract defines an Order struct containing six fields: buyer address, seller address, payment amount in Wei, delivery confirmation status, payment release status, and an order identifier. Three core functions govern the contract: createOrder locks buyer funds atomically on-chain upon order creation and emits an OrderCreated event; confirmDelivery, callable only by the original buyer, releases funds directly to the seller via payable transfer; and getOrder provides read-only order state for transparent auditability.

```
function createOrder(address _seller) external payable returns (uint256)
```

```
function confirmDelivery(uint256 _orderId) external
```

```
function getOrder(uint256 _orderId) external view returns (address, address, uint256, bool, bool)
```

These functions collectively implement a three-stage lifecycle: fund locking, delivery verification, and automatic settlement.

4.3. Comparison with traditional cross-border payment systems

Table 1. Comparison of traditional vs. blockchain payment systems

Dimension	Traditional SWIFT	This Blockchain DApp
Settlement Speed	3–5 business days	~15 seconds (Sepolia testnet)
Intermediaries	2–4 correspondent banks	None (contract executes autonomously)
Transaction Fees	2–5% of transfer value	Gas fees only (~\$0.01–\$2 on testnet)
Transparency	Opaque, siloed bank records	Public blockchain ledger
Counterparty Risk	High —depends on bank solvency	Reduced via escrow mechanism
Accessibility	Requires banking relationships	Accessible via wallet address
Auditability	Manual bank records	Real-time blockchain verification

The comparison (Table 1) demonstrates that the blockchain-based design directly addresses the core inefficiencies identified in Section II. By eliminating correspondent banks, the system reduces both settlement time and transaction costs substantially. The immutability of the Ethereum ledger replaces the need for trusted intermediaries, which is particularly significant for small enterprises that lack established banking relationships across borders.

4.4. Transaction workflow

The payment process consists of three stages. First, the buyer connects a MetaMask wallet and creates an order by entering the seller's address and payment amount. The funds are then locked in the smart contract escrow, and an OrderCreated event is recorded on-chain. Confirmation on the Sepolia test network usually takes around 15 seconds.

In the second stage, the seller checks the order status through the `getOrder` function, which returns details such as the buyer address, payment amount, and current order state. This allows the seller to confirm that payment has been secured before shipping the goods.

After delivery, the buyer confirms receipt using the assigned order ID. The smart contract verifies that the request is submitted by the original buyer before automatically releasing the escrowed funds to the seller. Both transactions are permanently stored on-chain and can be verified through Etherscan.

5. Implementation

5.1. Development environment and tools

The implementation was conducted using macOS with Visual Studio Code as the primary development environment. The backend smart contract was developed using Solidity ^0.8.28 and compiled via Hardhat v2.22.0.

The frontend was developed using HTML5 and JavaScript, with ethers.js v5.7.2 for blockchain interaction. Wallet authentication and transaction signing were handled through MetaMask. This project is version-controlled on GitHub and deployed via Vercel for frontend hosting, enabling HTTP-based access required for wallet injection.

5.2. Smart contract deployment

The smart contract was compiled using Hardhat and deployed to the Ethereum Sepolia test network via Remix IDE using the Injected Provider (MetaMask) configuration. Remix IDE was selected due to compatibility issues between Hardhat v3 and Node.js v24 environments. The deployed contract is publicly verifiable on Sepolia block explorers.

5.3. Frontend implementation and deployment

The frontend is implemented as a single-page application with four core functional modules: wallet connection, order creation, delivery confirmation, and order status query.

The wallet connection function uses Web3Provider to detect MetaMask. The `createOrder` function parses the ETH amount using `parseEther()` and extracts the `OrderCreated` event from the transaction receipt to display the assigned order identifier. The frontend was deployed to Vercel with the root directory set to the frontend folder. Live URL: <https://blockchain-payment-snowy.vercel.app>

5.4. End-to-end transaction test

A complete end-to-end transaction was successfully executed on the deployed system. Using two separate MetaMask accounts on Sepolia, Account 1 (buyer) initiated a `createOrder` transaction specifying Account 2 as the seller with 0.001 SepoliaETH. The transaction was confirmed on-chain within approximately 15 seconds and returned Order ID: 1. Subsequently, Account 1 called `confirmDelivery` with Order ID 1, triggering automatic transfer of 0.001 SepoliaETH from contract escrow to Account 2. The frontend displayed: Confirmed! Payment has been automatically released to seller. Both transactions are permanently recorded on the Sepolia blockchain and verifiable at <https://sepolia.etherscan.io> using the contract address.

5.5. Security and reliability analysis

Escrow Security: Buyer funds are locked within the smart contract upon order creation and are inaccessible to any external party until the predefined release condition is met.

This eliminates counterparty risk inherent in traditional wire transfers. The escrow is enforced at the Ethereum Virtual Machine level, meaning no administrative override is possible once the contract is deployed.

Access Control: The `confirmDelivery` function is restricted to the original buyer address recorded at order creation, preventing any third party from triggering fund release. The `getOrder` function is publicly accessible as a read-only call, ensuring transparency without compromising fund security. These controls are immutably encoded in the contract bytecode and cannot be modified after deployment.

Immutability and Auditability: Once deployed, the contract code and all transaction records are permanently stored on-chain and cannot be altered. Every transaction is recorded with a unique hash, block number, and timestamp, providing a tamper-proof audit trail verifiable through Etherscan without relying on any centralised institution.

6. Discussion

6.1. Technical challenges and resolutions

The most significant technical challenge was the dependency conflict between Hardhat v3 and Node.js v24.13.0. Hardhat v3 introduced ECMAScript Module format as its default module system, which conflicted with CommonJS-based dependencies including `hardhat-ethers` and `hardhat-chai-matchers`. Multiple resolution attempts failed, including reinstalling with legacy peer dependency flags and manually installing individual plugins. The resolution was to pivot from Hardhat to Remix IDE for contract deployment — a browser-based Solidity environment requiring no local dependency configuration. This experience illustrates an important practical lesson: in blockchain development, tool selection should be driven by task requirements rather than default toolchain assumptions.

6.2. MetaMask integration insights

A notable challenge arose when testing the frontend locally via the local file protocol. MetaMask's security model does not inject the wallet provider into pages loaded from the local filesystem, causing the Connect MetaMask button to produce no response. This was resolved by deploying to Vercel, which serves the application over secure HTTP — a requirement for MetaMask's provider injection. Additionally, the originally specified content delivery network for `ethers.js` returned a version incompatible with the latest MetaMask provider API. Switching to the Cloudflare distribution of `ethers.js` v5.7.2 resolved the issue. These two challenges are rarely documented in academic literature but are critical practical hurdles for any Web3 frontend deployment.

6.3. Reflections on blockchain for small enterprise payments

Although the system successfully completed escrow settlement within 15 seconds during testing, real deployment surfaced several practical constraints that simulation-based studies tend to overlook. Using volatile cryptocurrency instead of a stable payment asset creates exchange rate uncertainty that may limit suitability for real trade environments. In addition, wallet configuration and

transaction management remain difficult for many non-technical SME users. These findings indicate that technical performance alone is insufficient to guarantee adoption, especially when usability and financial stability remain unresolved.

7. Conclusion

This project examined whether blockchain technology could reduce transaction costs and settlement risks in cross-border supply chain payments for SMEs. By designing and deploying an Ethereum-based decentralised application, the study demonstrated that smart contract settlement can provide faster and more transparent payment processing compared with the traditional wire transfer model.

The results show that blockchain-based escrow settlement can reduce transaction time from several business days to around fifteen seconds while removing most intermediary-related costs. At the same time, practical testing also revealed several challenges that are often overlooked in simulation-based research. These include cryptocurrency price volatility, high gas fees during periods of network congestion, and the technical difficulty of wallet management for inexperienced users.

Future improvements could include integrating stablecoins to reduce exchange-rate risk, using Layer 2 networks to lower transaction fees, and simplifying wallet onboarding for SME users. Additional work could also explore oracle-based delivery verification to further automate payment release conditions. Overall, the project shows that blockchain technology has strong potential to improve cross-border payment processes for small enterprises, although further development is still needed before large-scale real-world adoption becomes practical.

References

- [1] World Bank. (2024). Remittance Prices Worldwide. Retrieved from <https://remittanceprices.worldbank.org>
- [2] Ayodele, E., Oye, M. A., Alimi, B. C., & Obitolu, S. B. (2025) Investigating blockchain-based smart contracts for cross-border payment settlement, regulatory compliance and risk reduction in international finance. *International Journal of Science and Research Archive*, 16(2), 052-073.
- [3] Fu, J., & Saad, N. H. M. (2023) Cross border e-commerce uses blockchain technology to solve payment risks. *International Journal on Recent and Innovation Trends in Computing and Communication*, 11(3s), 205-215. <https://doi.org/10.17762/ijritcc.v11i3s.6182>
- [4] Quan, J., Duan, Y., & Fu, Q. (2025) Optimizing international trade settlement systems through blockchain-driven tri-chain collaboration. *Sustainable Futures*, 10: 101304. <https://doi.org/10.1016/j.sfr.2025.101304>
- [5] Guo, H., & Liu, X. (2025) Exploring trust dynamics in finance: The impact of blockchain technology and smart contracts. *Humanities and Social Sciences Communications*, 12(1), 1-10.
- [6] Yigit, E., & Dag, T. (2024) Improving supply chain management processes using smart contracts in the ethereum network written in solidity. *Applied Sciences*, 14(11), 4738.
- [7] Garg, R. (2022) Ethereum based smart contracts for trade and finance. *International Journal of Economics and Management Engineering*, 16(11), 619-629.

Appendix A: system screenshots

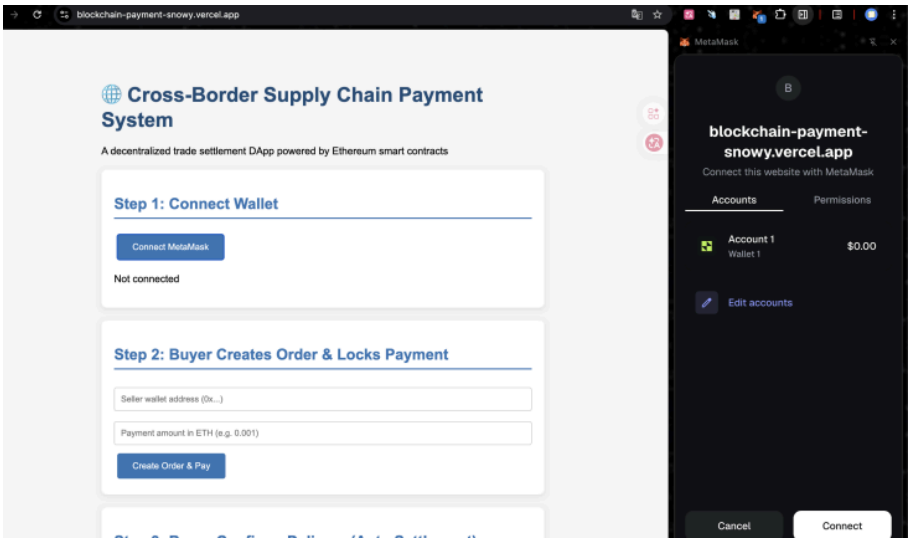


Figure A1. Live DApp interface and MetaMask wallet connection

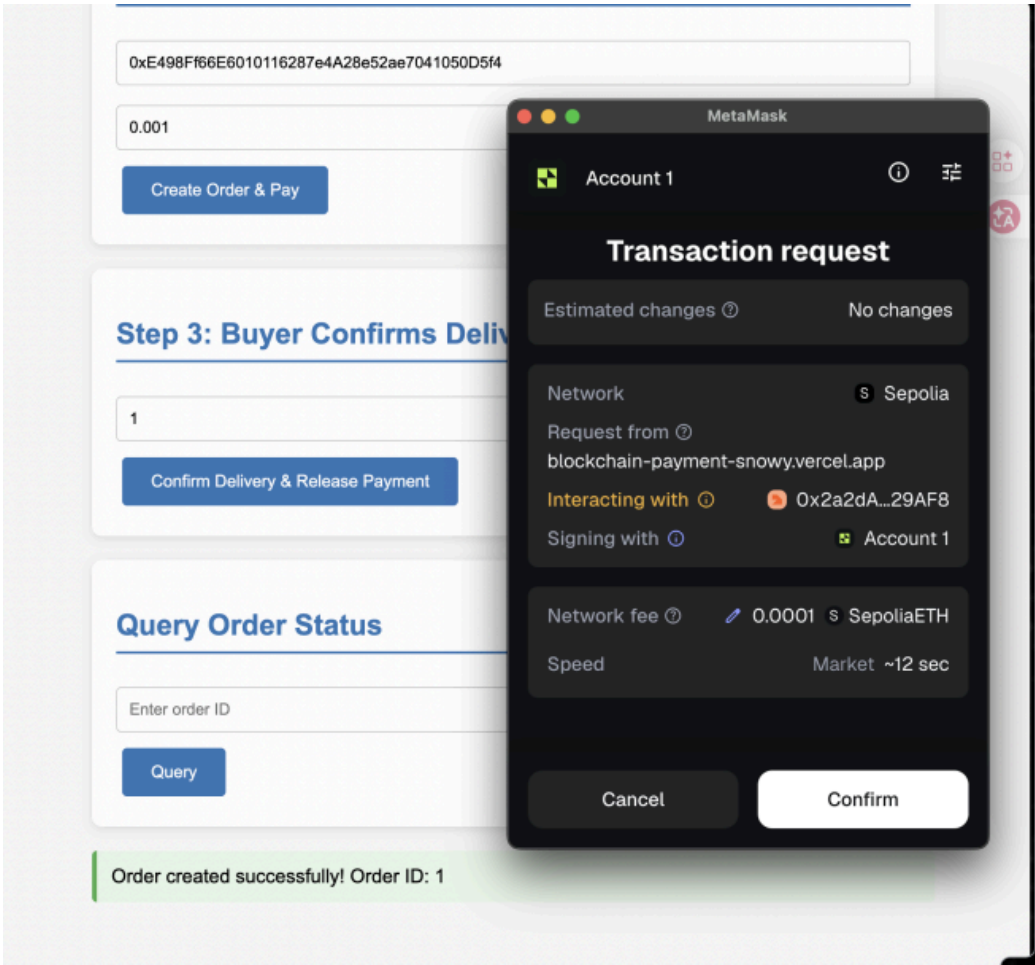


Figure A2. Order creation, smart contract interaction and successful payment confirmation