Blockchain-Based Upgradeable Cryptocurrency Credit Card System

Gaurav Acharya*, Sivaselvan Natarajan*
Manipal Institute of Technology,
Manipal Academy of Higher Education,
Manipal, India
gauravacharya1201@gmail.com, siva.selvan@manipal.edu

***Abstract*—Cryptocurrency is gaining popularity all over the world because it is not only used as an investment opportunity but also as a medium of transaction. With this, it is imperative that we have a credit card system that effectively manages purchases made with cryptocurrencies on credit. To achieve this, we have come up with a novel solution in this paper, a blockchain-integrated credit card system that enables decentralized processing of cryptocurrency transactions with embedded logic for credit usage, automated rewards, trust-based user profiling, and credit limit management. With the system deployed as an upgradeable smart contract on the blockchain network, the system supports modular deployment, thereby it can meet future business and regulatory requirements. The system improves transparency and operational latency by eliminating intermediaries and embedding the business logic directly into contract code, and supports maintainability through upgradeable patterns. A proof-of-concept implementation for the proposed system for ETH cryptocurrency has been developed and validated through extensive testing on Ethereum testnets such as Cancun, Mainnet fork, and Sepolia. The experimental results show that the transaction fee for the most intensive contract deployment operation itself is 0.004186 ETH. This demonstrates the feasibility of the system we developed.**

***Index Terms*—Blockchain, Cryptocurrency, Credit Card System, Decentralized Finance (DeFi), Ethereum, Upgradeable Contracts.**

I. Introduction

Cryptocurrency is gaining wider attention across the globe, with around 741 million people using it either for investment or transactions purpose in 2025 [1]. Credit card usage worldwide has seen a substantial increase. Particularly, the increase in usage is perceived in North America and Asia. In the US, the number of credit cards in circulation by the end of 2024 was 942 million. The total spending for credit cards for this year reached US$ 6.136 trillion [2]. In India, the number of active credit cards in the late 2024 was approximately 108 million and shows consistent growth [3]. According to the Reserve Bank of India report [4], the number of credit card transactions in India in 2024 was 4.47 billion, the transaction value amounting to INR 20.4 lakh crore (approximately 20 trillion). Credit card transactions added up to 2.66 billion in the first half of 2025.

Even with increased usage, the credit card system still faces major problems, where customers and merchants have to deal with a list of charges such as annual fees, cash advance fees, late payment penalties, and transaction fees. In particular, high transaction fees are a great concern since they are spent on every transaction, affecting both merchants and customers. In the US, transaction fees typically range from 1.1% to 3.15% [5]. In India, merchants pay from 1.5% to 3.5% per transaction [6]. Furthermore, transactions through credit cards are generally slow as the payment clearance process can be prolonged and involve many intermediaries, which will increase the risk of delays.

Despite these challenges, there is an opportunity for new solutions to improve efficiency by eliminating intermediaries and reduce the different costs in credit card systems without compromising transparency, trust, and security in such systems. This is where blockchain technology comes to light. Blockchain is decentralized, unalterable, secure, and transparent. A blockchain-based credit card system provides more transparency, security, and trust through decentralization, while positively influencing transaction efficiency and costs through blockchain smart contracts. In this paper, we explore how such a system could be realized on the blockchain network and how we can transact in cryptocurrency instead of fiat currency.

We made the following contributions in this paper:

- A blockchain-based credit card scheme for cryptocurrencies with a novel feature namely smart contract-driven system upgradeability to meet evolving business and regulatory requirements and credit functionalities like automated rewards, customer trustworthiness, and credit limit adjustment has been proposed.
- A proof-of-concept (PoC) implementation for the proposed system for ETH cryptocurrency using OpenZeppelin upgradeable smart contract library is developed.
- PoC has been evaluated in Cancun, Mainnet Fork, and Sepolia testnets. The transaction and execution costs of the different contract operations in these testnets are obtained. In addition, the upgradeable version of PoC is benchmarked with the immutable counterpart to highlight the significance of upgradeability in the proposed system.

The rest of the paper is organized as follows: Section II provides a literature survey of related work in blockchain-based payment and credit systems. Section III details the pro-

*The authors contributed equally to this work.

posed methodology and system design. Section IV discusses the results and analysis of the proposed approach. Finally, Section V concludes the paper and outlines the directions for future work.

II. Literature Review

Blockchain has been increasingly used in finance in recent years as evidenced by the recent studies [7] [8]. The use of blockchain technology in payment systems and credit card transaction networks has attracted significant interest as it addresses critical issues such as fraud, settlement delays, high transaction costs, data breaches, and traceability.

The research work by Godfrey-Welch [9] was the first of its kind to use blockchain in payment card systems. It discussed the existing payment card workflow and several ongoing issues including dependency on multiple intermediaries, maintenance of duplicated ledgers, high transaction fees, and increased exposure of sensitive cardholder data. The work tried to address some of these issues by devising a blockchain-based framework.

Pushpa Raj *et al.* [10] proposed a "blockchain-based framework to manage procurement, traceability, and advance cash credit within supply chain finance systems". The work models end-to-end procurement workflows using smart contracts. The framework demonstrates feasibility of smart contract–driven credit automation. However, it largely assumes static contract logic and does not address how long-term contract upgrades and regulatory changes could be handled in practice.

The use of blockchain technology to improve "credit management in commercial banking" was explored in [11]. By introducing a shared blockchain platform with smart contracts, the approach aims to improve data integrity, traceability, and collaboration across credit-related operations. However, the work remains conceptual and lacks implementation-level details. Moreover, the absence of discussion on contract upgrades limits its applicability to dynamic banking environments.

De Silva *et al.* [12] presented a "cryptocurrency-based card payment protocol built on payment channel networks". It draws inspiration from Bitcoin's Lightning Network and deploys transactions through smart cards. The proposed system focuses on achieving fast settlement and low transaction costs. Although the protocol addresses performance limitations of on-chain payments, it functions primarily as a prepaid or debit-style system. Core credit card features are not considered. Additionally,the reliance on Bitcoin-specific infrastructure restricts flexibility and long-term extensibility.

Researchers in [13] introduced a "hybrid cryptocurrency payment architecture for e-commerce platforms". The three-layer design improves resistance to common web-based attacks and reduces transaction fees in real-world deployments. The solution is primarily oriented towards transaction processing than credit functionality. Features such as credit limits, billing cycles, or contract evolution mechanisms are not addressed, limiting its suitability for credit-based financial services.

Kuren and Al-Turjman [14] developed a "blockchain-based online payment system", demonstrated through a university fee payment use case. The system emphasizes decentralization and immutability while offering an accessible user interface. While effective for secure payment transfers, the framework is limited to basic transaction execution and does not support advanced credit-related features. Moreover, the lack of structured smart contract design for upgrades raises concerns about maintainability as the system requirements change.

A blockchain-enabled framework that integrates ethereum smart contracts with the metering devices to automate usage tracking and payments using stablecoins was proposed in [15]. The system demonstrates feasibility through performance evaluation and is well suited for pay-per-use scenarios. However, it does not address broader consumer credit functionalities. In addition, the absence of contract versioning and upgrade strategies limits adaptability to evolving pricing models or regulatory constraints.

VeriCred, an "automated credit scoring framework" was introduced in [16]. The approach combines machine learning with blockchain-based auditing to enhance transparency and trust. It strengthens governance and explainability in credit scoring. In the approach, blockchain usage is confined to audit and compliance purposes, and upgrade considerations for evolving models or policies are not discussed.

Lin *et al.* [17] presented SecurePay, a "blockchain-based payment framework for platform economies and programmable Central Bank Digital Currency (CBDC) environments". The system addresses low-latency transaction processing and is also effective in addressing platform-level payment risks. However, the system's reliance on CBDC infrastructure and lack of discussion on smart contract evolution further limit its applicability to long-lived, adaptable payment and credit solutions.

TABLE I. Summary of literature review

Scheme	Focus area	Relevance; Limitation(s)
[10]	Blockchain-based supply chain finance with automated advance cash credit and traceability	Credit automation via smart contracts; lacks support for contract upgradeability
[11]	Blockchain-enabled credit management in commercial banking	Demonstrates credit transparency; remains conceptual with no evolving credit logic
[12]	Fast cryptocurrency card payments using payment channels	Targets debit-style payments; excludes credit features and issuer controls
[13]	Secure cryptocurrency payments for e-commerce platforms	Focuses on payment acceptance; no on-chain credit or upgrade mechanisms
[14]	Decentralized online payment system	Supports basic payments only; lacks advanced credit card functionality
[15]	Blockchain-driven utility payments using stablecoins	Limited to utility payment; does not support flexible credit models
[16]	Blockchain-audited automated credit scoring framework	Improves credit assessment transparency; no payment or credit system
[17]	Blockchain-based payment framework for platform economies and CBDC	Addresses platform payments; not suitable for evolving credit payment systems

Table I presents a summary of recent and relevant sources ([10] – [17]) for our literature survey. Together, these studies provide a comprehensive and technically sound basis for the development of blockchain-based card payment systems with credit functionalities characterized by system upgradeability, cryptographic security, real-time settlements, low transaction costs, and traceability while acknowledging the existing infrastructural and regulatory hurdles.

Therefore, in this work, we propose a credit card system for cryptocurrencies based on upgradeable blockchain smart contracts that can adapt well to evolving business, legal, and regulatory requirements.

III. Methodology

This section discusses the design of the proposed upgradeable cryptocurrency credit card system using blockchain technology. The system uses smart contracts to manage credit card functionalities, automate rewards, and automatically adjust the credit limit based on user's behaviour.

A. System Architecture

In Fig. 1, the high-level architecture for the cryptocurrency credit card system is presented. The various components and interactions among the components are shown. The system is realized through smart contracts deployed on a blockchain network.

The architecture consists of the following components:

- **Smart Contract Deployer:** The smart contract deployer deploys the smart contract on the blockchain network. It deploys the proxy and implementation contracts. It is responsible for managing access control for different kinds of actors in the system. Also, it has administrative privileges such as retrieving customers' account details and modifying their credit limits.
- **Proxy Contract:** This is the smart contract through which various actors interact with the implementation contract. The proxy contract contains the state variables.
- **Implementation Contract:** This smart contract contains the entire contract logic. It is upgradeable in nature so that the system can meet the evolving business needs. The design of this contract can be found in Section III-B.
- **Customer:**
  - **Credit account structure:** Each customer is associated with a credit account containing name, credit limit, credit used, total reward points, and a trust indicator.
  - **Debit account structure:** Each customer also has a debit account with name and debit balance.
- **Validator:** The primary role of a validator is to validate and confirm transactions by either the proof of work or energy-efficient proof of stake consensus. Once a validator successfully verifies a transaction, they add the new block of verified transactions to the blockchain, ensuring the ledger remains consistent across the network.
- **Provider:** The provider allows customers to purchase various products and services using their cryptocurrencies on credit.

The interactions among the various components are described below:

1) The smart contract deployer deploys implementation contract on the blockchain network.
2) The smart contract deployer deploys proxy contract on the blockchain network. The proxy contract points to the current implementation contract. To add new features to the system to meet the evolving business requirements, the smart contract deployer deploys a new implementation contract and updates the proxy contract to point to the new implementation contract.
3) A customer makes purchases, pays the provider for products with its credit account by interacting with the proxy contract through a blockchain transaction.
4) A customer pays the dues to its credit wallet with its debit account by interacting with the proxy contract.
5) The smart contract deployer retrieves the credit and debit account details of customers through interaction with the proxy contract.
6) The smart contract deployer adjusts the credit limit of a trustworthy customer through interaction with the proxy contract.

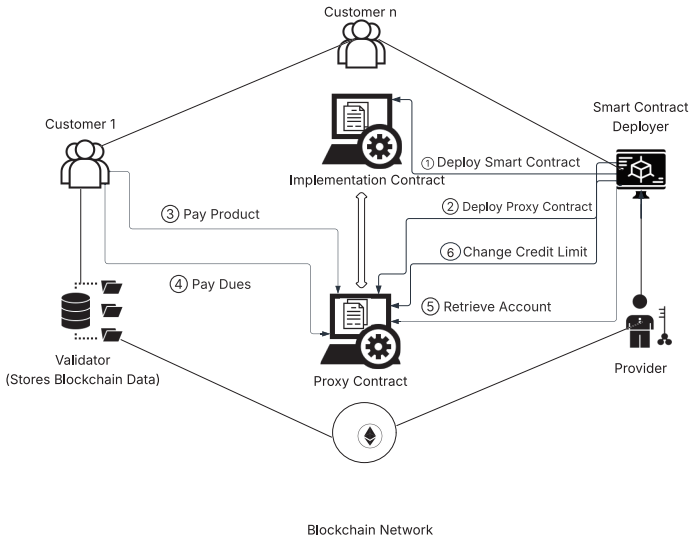

Fig. 1. High-level architecture of the blockchain-based up-gradeable cryptocurrency credit card system

B. Smart Contract Design

1) **Initialization:** The `initialize()` function sets up customer accounts with initial cryptocurrency credit and debit balances, credit limits, and reward points.
2) **Product Payment:** The `payproducts()` function allows customers to make payments for various products and services using their credit account. The function checks the available credit limit, processes the payment, and gives appropriate rewards or surcharge waiver benefits based on the product category.

3) **Dues Payment:** The `paydues()` function allows customers to pay their credit card dues using their debit account. It checks if the customer has sufficient debit balance, updates the credit and debit balances, and adjusts the trust indicator based on the completion of payment of dues.
4) **Account Retrieval:** The `retrievecaccount()` and `retrievedaccount()` functions allow the contract owner to view customers' credit and debit account details respectively.
5) **Credit Limit Adjustment:** The `changecreditlimit()` function permits the contract owner to increase a customer's credit limit provided it has maintained trustworthiness marked by the trust indicator.

By using blockchain smart contracts, the proposed methodology ensures transparent, automated, and tamper-resistant management of cryptocurrency credit card transactions. The proposed design overcomes the limitations in conventional credit card systems by eliminating the intermediaries and associated fees, maintaining transparency of transactions, and providing real-time rewards and credit limit adjustment. Above all, the proposed approach offers system upgradeability to add new credit functionalities and security patches as per evolving business and legal requirements through the use of upgradeable smart contract paradigm.

IV. Results Analysis

We developed PoC (Algorithms 1–5) for the proposed system particularly for ETH cryptocurrency using the OpenZeppelin's upgradeable smart contract paradigm [18] consisting of an ERC proxy contract and implementation contract. Experiments were conducted in Cancun, Mainnet fork, as well as the Sepolia Ethereum testnet to analyze the smart contract performance under realistic execution scenarios. We also benchmarked the performance and features of the upgradeable smart contract implementation of PoC with the non-upgradeable one to highlight the significance and effectiveness of upgradeability in the proposed system.

A. Secure Contract Upgrades

The PoC uses `OwnableUpgradeable` module from the OpenZeppelin library to allow only the contract deployer (contract owner) to perform authorized contract upgrades.

B. Smart Contract Algorithms

The smart contract functionalities presented in Section III-B are realized in Algorithms 1–5. In Algorithm 1, the contract is initialized as upgradeable using the OpenZeppelin's universal upgradeable proxy standard. The cryptocurrency debit balance, credit limit, trustworthiness, and reward points are initialized for all customers in the system. In Algorithm 2, the pseudocode for `payproducts()` function is presented. It verifies if the customer's credit balance is sufficient to pay for products. If yes, it transfers cryptocurrency from the customer's credit wallet to the provider. Also, it computes the applicable rewards or waiver for the customer's payment towards purchase of products. The rewards so computed can be credited to the customer's accumulated rewards after successful dues payment. Algorithm 3 presents the pseudocode for `paydues()` function. It negates the customer's trustworthiness if the debit balance is insufficient to pay the dues. Algorithm 4 provides the contract owner with the credit and debit account details of customers upon invocation. Algorithm 5 verifies the customer's trustworthiness based on its history of paying the credit dues. If found trustworthy, the algorithm lets the contract owner double the customer's credit limit. Table II presents the time complexity of these algorithms. The `initialize()` function takes $\mathcal{O}(n)$ time complexity while the remaining functions run in constant time.

Algorithm 1: Initialization – `initialize()` function

Input: —

```
// Setting contract deployer as the
   contract owner
Ownable_init(msg.sender);
// Setting the contract upgradeable
UUPSUpgradeable_init();
// Initialize credit and debit
   accounts
credit_limit ← 1 ETH;
credit_used ← 0;
tot_reward ← 1000 points;
customer_trust ← true;
debit_balance ← 1 ETH;
```

Algorithm 2: Product Payment – `payproducts()` function

Input: $product\_type, payment\_amount$ (in ETH), $provider\_address$

```
if credit_available < payment_amount then
    revert "Insufficient credit";
end
credit_used ← credit_used + payment_amount;
credit_limit ← credit_limit − payment_amount;
provider_address.transfer(payment_amount);
if product_type is retail then
    reward ← (payment_amount/1 ETH) × 10;
    tot_reward ← tot_reward + reward;
else
    waiver ← (payment_amount/1 ETH)×1 Gwei;
    credit_limit ← credit_limit + waiver;
end
```

C. Experimental Evaluation of Proof-of-Concept

In this section, we evaluate the implementation contract viz., Upgradeable Credit Contract (UCC) and ERC proxy contract in our PoC implementation in the Cancun, Mainnet Fork, and

Algorithm 3: Dues Payment – `paydues()` function

Input: $payment\_amount, credit\_address$
if $debit\_balance < credit\_used$ **then**
  `// Mark customer as untrustworthy`
  $customer\_trust \leftarrow false$;
end
if $debit\_balance < payment\_amount$ **then**
  revert "Insufficient balance";
end
$credit\_used \leftarrow credit\_used - payment\_amount$;
$credit\_limit \leftarrow credit\_limit + payment\_amount$;
$debit\_balance \leftarrow debit\_balance - payment\_amount$;
$credit\_address.transfer(payment\_amount)$;

Sepolia testnets. We acquired Sepolia ETH required for smart contract analysis in Sepolia testnet from the authenticated Sepolia faucet hosted by Google Cloud [19]. Experiments were conducted on these test networks using real and simulated blockchain transaction data. The Remix IDE and MetaMask tools were used. The metrics chosen for evaluation of the smart contracts' performance are transaction and execution costs (gas) for Remix Cancun and Mainnet Fork, and transaction fee (ETH) for the Sepolia testnet. Finally, we compared the upgradeable implementation of our PoC with the non-upgradeable version in terms of contract state (lost or retained) and cost factors.

Algorithm 4: Account Retrieval – `retrievecaccount()`, `retrievedaccount()` functions

Input: $credit\_address, debit\_address$
if $msg.sender \neq contract\_owner$ **then**
  revert "Unauthorized";
end
return $credit\_account, debit\_account$;

Algorithm 5: Credit Limit Adjustment – `changecreditlimit()` function

Input: $credit\_address$
if $msg.sender \neq contract\_owner$ **then**
  revert "Unauthorized";
end
if $customer\_trust = false$ **then**
  revert "Untrustworthy user";
end
`// Double credit limit`
$credit\_limit \leftarrow credit\_limit \times 2$;

1) Remix Cancun and Mainnet Fork Evaluation: We evaluated the UCC and ERC proxy contracts in the Cancun configuration and Mainnet Fork. The transaction and execution costs of different contract operations in these testnets are presented in the Tables III and IV. Overall, the tables present similar results. In connection with UCC, we included the costs only for the `initialize()` function because this is the only function of UCC that is directly invoked, and the other functions are invoked through the ERC proxy contract. The initialization operation in UCC incurred the highest transaction cost in both the Cancun and Mainnet Fork testnets with 2,782,090 and 2,688,004 gas units respectively. Whereas, the transaction for initialization operation in ERC proxy contract consumed 509,770 and 505,699 gas in these testnets. The cost for initialization operation covers both the contract deployment and initialization costs since the contract is initialized soon after its deployment in the environment.

TABLE II. SMART CONTRACT FUNCTIONS – TIME COMPLEXITY

Contract function	Time complexity
`1.initialize()` (Invoked only once)	$\mathcal{O}(n)$*
`2.payproducts()` `3.paydues()` `4.retrievecaccount()`, `retrievedaccount()` `5.changecreditlimit()`	$\mathcal{O}(1)$

* n = number of customers.

The transactional functions like i) `payproducts` used 70,714 gas units per invocation in both Cancun and Mainnet Fork, ii) `paydues` consumed 48,033 and 37,733, and iii) `changecreditlimit` consumed 36,690 and 36,688 gas units. In the Cancun configuration, the execution costs of these functions account for approximately 69.5%, 65.4%, and 41.6% of their transaction costs. Whereas, in Mainnet Fork, the execution costs of transactional functions account for 69.4%, 43.2%, and 41.6% of the transaction costs.

Read-only functions like `retrievecaccount` and `retrievedaccount` incurred no or zero transaction cost. However, they incurred execution costs of reasonable figures.

2) Evaluation in Sepolia Testnet: Table V demonstrates the transaction fee (ETH) of different contract operations in the Sepolia testnet. Deployment of ERC Proxy Contract consumed 2,782,102 gas units and incurred a transaction fee of 0.004186 ETH. Whereas, the UCC `initialize` function used up 2,769,502 gas resulting in a transaction fee of 0.004157 ETH. Transactional functions like `payproducts` and `paydues` required 38,514 and 37,733 gas units. The transaction fees incurred are 0.0000578 and 0.0000567 ETH respectively. The function `changecreditlimit` consumed 36,690 gas units incurring 0.0000551 ETH. The functions `retrievecaccount` and `retrievedaccount` do not require any transaction fee. The gas price is about 1.5 Gwei across the contract operations. When it comes to the gas fees of different contract operations, the priority fee is constant throughout with minor variations in the base and max fees.

D. Benchmarking

In this section, we benchmarked the upgradeable implementation of PoC with the traditional immutable version to

TABLE III. GAS USAGE FOR THE SMART CONTRACT FUNCTIONS IN CANCUN

	ERC Proxy Contract						UCC
	initialize	payproducts	paydues	retrievecaccount	retrievedaccount	changecredit limit	initialize
Transaction Cost (gas)	509,770	70,714	48,033	**		36,690	2,782,090
Execution Cost (gas)	430,094	49,142	31,401	20,210	13,429	15,258	2,536,006

** Zero transaction cost.

TABLE IV. GAS USAGE FOR THE SMART CONTRACT FUNCTIONS IN MAINNET FORK

	ERC Proxy Contract						UCC
	initialize	payproducts	paydues	retrievecaccount	retrievedaccount	changecredit limit	initialize
Transaction Cost (gas)	505,699	70,714	37,733	**		36,688	2,688,004
Execution Cost (gas)	426,023	49,142	16,301	20,210	13,429	15,256	2,448,108

** Zero transaction cost.

TABLE V. ETHER USAGE FOR THE SMART CONTRACT FUNCTIONS IN SEPOLIA TESTNET

	ERC Proxy Contract					UCC
	initialize	payproducts	paydues	retrievecaccount & retrievedaccount	changecredit limit	initialize
Tx Fee (ETH)	0.004186	0.0000578	0.0000567	**	0.0000551	0.004157
Gas Price (Gwei)	1.5046	1.5014	1.5015	—	1.5008	1.5043
Gas Alloted	2,805,349	41,092	38,099		37,053	2,800,180
Gas Usage	2,782,102 (99.17%)	38,514 (93.73%)	37,733 (99.04%)		36,690 (99.02%)	2,769,502 (98.91%)
Gas Fee	Base: 0.0046 Max: 1.5063 Priority: 1.5	Base: 0.0014 Max: 1.5019 Priority: 1.5	Base: 0.0014 Max: 1.5019 Priority: 1.5		Base: 0.0008 Max: 1.5013 Priority: 1.5	Base: 0.0046 Max: 1.5061 Priority: 1.5

** Zero transaction fee.

TABLE VI. BENCHMARKING THE UPGRADEABLE SMART CONTRACT WITH ITS TRADITIONAL IMMUTABLE IMPLEMENTATION

	Contract State / Contract Address	Costs				
		Cancun		Mainnet Fork		Sepolia Testnet
		Transaction Cost (gas)	Execution Cost (gas)	Transaction Cost (gas)	Execution Cost (gas)	Tx Fee (ETH)
Traditional Implementation (Redeploy operation)	LOST	1,565,315 UMC 1 U: 67,050 10 U: 670,500 100 U: 6,705,000 **Total: 8,270,315**	1,383,141 UMC 1 U: 23,906 10 U: 239,060 100 U: 2,390,600 **Total: 3,773,741**	1,447,663 UMC 1 U: 66,272 10 U: 662,720 100 U: 6,627,200 **Total: 8,074,863**	1,273,429 UMC 1 U: 23,268 10 U: 232,680 100 U: 2,326,800 **Total: 3,600,229**	0.002111 UMC 1 U: 0.00005725 10 U: 0.0005725 100 U: 0.005725 **Total: 0.007836**
Upgradeable Contract (Upgrade operation)	RETAINED	UCC: 2,574,611 ERC Proxy: 38,607 UMC 1/10/100 U: 0 **Total: 2,613,218**	UCC: 2,342,381 Proxy: 16,907 UMC 1/10/100 U: 0 **Total: 2,359,288**	UCC: 2,665,720 Proxy: 38,607 UMC 1/10/100 U: 0 **Total: 2,704,327**	UCC: 2,427,478 Proxy: 16,907 UMC 1/10/100 U: 0 **Total: 2,444,385**	UCC: 0.003998 Proxy: 0.000057 UMC 1/10/100 U: 0 **Total: 0.004055**

** UMC – User Migration Cost **;** 1/10/100 U – 1/10/100 Users

underline the importance of upgradeability in the proposed Decentralized Finance (DeFi) driven cryptocurrency credit system. Table VI presents benchmarking in terms of contract state, contract address, and costs. The cost comparisons are conducted across the Cancun, Mainnet Fork, and Sepolia testnets. The proposed DeFi-driven smart contract-based cryptocurrency credit system requires contract upgrades to meet the evolving business needs. Using traditional immutable contracts for this purpose requires complete redeployment of the contract for every upgrade resulting in complete loss of contract state and address. Consequently, blockchain transactions for migrating users (user refers to "Customer" in the proposed system in Fig. 1) to the new contract instance are to be initiated to retain the present contract state. These transactions incur

user migration costs (UMC). This is why we use upgradeable smart contracts in the proposed system where the contract upgrades could be done without losing the contract state.

The overall costs covering contract deployment cost and the resulting UMC for traditional immutable and upgradeable versions of PoC during a contract upgrade are shown in Table VI. The costs incurred across the three testnets are presented. It can be seen that the deployment cost (one-time cost) during an upgrade operation is comparatively high in the upgradeable version in all the three testnets. However, the UMC (repeat cost) is zero irrespective of the number of users. Whereas, the UMC in traditional immutable version during an upgrade operation increases with the increase in the number of users though the redeployment cost is relatively less. This trend of UMC leads to high and rising overall costs in traditional immutable version compared to the upgradeable one. To elaborate, the overall transaction and execution costs for the traditional version in Cancun are 8,270,315 and 3,773,741 gas units respectively. We considered the number of users to be 100 in the overall cost computation. Also, we included only the costs of `payproducts` and `paydues` operations in the per-user cost computation. The above figures for overall transaction and execution costs are high compared to those for upgradeable version viz., 2,613,218 and 2,359,288 gas. The overall costs follow similar trend in Mainnet Fork as well. In Sepolia testnet, the overall transaction fee for traditional immutable version is 0.007836 ETH which is almost double the overall fee of upgradeable one (0.004055 ETH).

Overall, the reasonable transaction and execution cost figures for PoC across the testnets demonstrate feasibility of the proposed system. Feasibility is further endorsed by the advantages the upgradeable version of PoC brings in terms of contract state and UMC compared to the immutable version.

V. Conclusion

The implementation of a blockchain-based credit card system utilizing upgradeable smart contracts establishes a robust, decentralized, and adaptable framework for secure and autonomous credit-related operations. The design covers core credit mechanisms such as credit issuance, dues settlement, rewards computation, and credit limit adjustment directly within the smart contract layer, thereby reducing dependency on centralized intermediaries and mitigating the associated risks. Access control and authorized contract upgrades via role-based permissions ensure integrity, while the modular architecture supports extensibility and contract evolution without disrupting system continuity.

Evaluation across diverse ethereum environments demonstrates architecture efficiency and feasibility of the proposed system. Benchmarking proxy-based upgradeable implementation of the proposed system with the conventional immutable version reveals the importance of system upgradeability in highly dynamic DeFi-driven credit card applications.

Further research may investigate the integration of advanced credit scoring framework and dynamic credit limit adjustment into the proposed system to make it comprehensive and more usable in real-world credit or payment use cases for cryptocurrencies and CBDCs. Also, exploring layer-2 scaling solutions such as rollups or sidechains can extend the system's reach to real-world financial ecosystems with greater scalability, faster transactions, and lower fees.

References

[1] Crypto.com, "Global cryptocurrency ownership reaches 741 million in 2025," [Online]. Available: https://crypto.com/en/company-news/global-cryptocurrency-ownership-reaches-741-million-in-2025. Accessed: April 01, 2026.

[2] "JP Morgan tops Nilson report ranking of US credit card issuers," [Online]. Available: https://www.globenewswire.com/news-release/2025/03/06/3038338/0/en/JP-Morgan-Tops-Nilson-Report-Ranking-of-US-Credit-Card-Issuers.html. Accessed: December 09, 2025.

[3] K. Author, "Credit card usage in India: 2025 trends and consumer behaviour," [Online]. Available: https://gokiwi.in/blog/credit-card-usage-in-india-2025-trends-and-consumer-behaviour/. Accessed: December 09, 2025.

[4] P. Debnath, "Credit card transactions increase as debit card usage continues to decline: RBI report," [Online]. Available: https://www.outlookmoney.com/banking/credit-card/credit-card-transactions-increase-as-debit-card-usage-continues-to-decline-rbi-report/. Accessed: December 09, 2025.

[5] J. Caporal, "Average credit card processing fees and costs in 2025," [Online]. Available: https://www.fool.com/money/research/average-credit-card-processing-fees-costs-america/. Accessed: December 10, 2025.

[6] J. Kaur, "Credit card processing fees & charges August 2025," [Online]. Available: https://cardgrid.in/blog/credit-card-processing-fees-charges/. Accessed: December 10, 2025.

[7] T. Tanchangya, T. Sarker, J. Rahman, M. S. Islam, N. Islam, and K. O. Siddiqi, "Mapping blockchain applications in FinTech: A systematic review of eleven key domains," *Information*, vol. 16, no. 9, p. 769, 2025.

[8] G. D. Sharma, A. K. Tiwari, R. Chopra, and D. Dev, "Past, present, and future of blockchain in finance," *Journal of Business Research*, vol. 177, p. 114640, 2024.

[9] D. Godfrey-Welch, R. Lagrois, J. Law, R. S. Anderwald, and D. W. Engels, "Blockchain in payment card systems," *SMU Data Science Review*, vol. 1, no. 1, art. 3, 2018.

[10] P. V. R. Pushpa Raj, S. K. Jauhar, M. Ramkumar, and S. Pratap, "Procurement, traceability and advance cash credit payment transactions in supply chain using blockchain smart contracts," *Computers and Industrial Engineering*, vol. 165, p. 107948, 2022.

[11] X. Huang, "Application of blockchain technology in credit management of commercial banks," *SSRN*, 2024. https://dx.doi.org/10.2139/ssrn.5000388

[12] A. De Silva, S. Thakur, and J. Breslin, "Card payment protocol for cryptocurrencies with payment channel network," *Distributed Ledger Technologies: Research and Practice*, vol. 3, no. 3, Art. no. 19, pp. 1–30, 2024.

[13] L. Navarro, J. Mansilla-Lopez, and C. Cipriano, "Technological model for cryptocurrency payments in e-commerce," in *Proc. 20th Int. Conf. on Web Information Systems and Technologies (WEBIST)*, 2024.

[14] H. Kuren and F. Al-Turjman, "A simple online payment system using blockchain technology," in *Advanced Studies in Complex Systems, Computational Intelligence and Blockchain in Complex Systems*, pp. 25–37, 2024.

[15] A. D. J. William, S. Rajendran, P. Pranam, and Y. Berry, "Blockchain technologies: Smart contracts for consumer electronics data sharing and secure payment," *Electronics*, vol. 12, no. 1, 2023.

[16] X. Dong, F. Yang, X. Dai, and Y. Qiao, "A novel blockchain architecture for secure and transparent credit regulation," *Applied Sciences*, vol. 15, no. 23, 2025.

[17] J. Lin, M. Liu, S. Li, and X. Wang. "SecurePay: Enabling secure and fast payment processing for platform economy." arXiv preprint arXiv:2505.17466 [2025]. https://doi.org/10.48550/arXiv.2505.17466

[18] OpenZeppelin, "Upgradeable variant of OpenZeppelin contracts," [Online]. Available: https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable. Accessed: May 01, 2025.

[19] Google Cloud, "Ethereum Sepolia faucet," [Online]. Available: https://cloud.google.com/application/web3/faucet/ethereum/sepolia. Accessed: May 15, 2025.