iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** XII **Month of publication:** December 2025

DOI: https://doi.org/10.22214/ijraset.2025.75841

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue XII Dec 2025- Available at www.ijraset.com

Unified Fiat and Crypto Payment Platform

Sanjivani Adsul[1], Tanishka Pimple[2], Tejas Sarade[3], Varun Sahu[4], Kedar Vartak[5]
Artificial Intelligence and Data Science, Vishwakarma Institute of Technology, Pune, India

Abstract: ***The evolution of digital payment systems has revolutionized how people transact in modern economies. In India, Unified Payments Interface (UPI) has enabled a massive leap forward in convenience and digital finance adoption. However, cryptocurrency transactions have not kept pace due to infrastructural and usability challenges. Users often navigate multiple platforms, wallets, and address formats, making crypto less accessible. This paper presents UniPay, a unified digital payment solution that integrates fiat (like INR via UPI) and cryptocurrency transactions within a single platform. By bridging this gap, UniPay aims to offer a seamless, secure, and interoperable transaction experience for users and merchants, enhancing financial inclusivity.***

Keywords: Cryptocurrency, Digital Payments, Blockchain, Interoperability, FinTech, Crypto Wallet, Crypto-fiat Integration

I. INTRODUCTION

With increasing digital transformation, the demand for seamless and inclusive financial platforms is growing rapidly. The introduction of UPI in India has brought unprecedented ease and speed to fiat currency transactions. Conversely, cryptocurrencies, despite their advantages in decentralization and transparency, remain underutilized due to their technical complexity and regulatory uncertainties. Managing multiple wallets, performing manual conversions, and dealing with long alphanumeric addresses create friction, especially for average users. UniPay aims to address these issues by combining the simplicity of fiat digital payments with the versatility of crypto transactions. It allows real-time fiat-to-crypto and crypto-to-fiat conversions while maintaining regulatory compliance, intuitive UX, and high security.

II. LITERATURE REVIEW

Numerous studies have explored both centralized digital payment systems and decentralized cryptocurrency frameworks, highlighting the unique advantages and challenges of each. Sarkar et al. (2021) examined the role of UPI in enhancing digital financial inclusion in India and noted its transformative impact on retail transactions and peer-to-peer payments [1]. Similarly, Gupta and Jain (2020) analyzed UPI's security architecture and concluded that its simplicity combined with multi-layered authentication has led to its wide adoption [2]. In contrast, Nakamoto's seminal paper on Bitcoin introduced blockchain as a trustless, peer-to-peer system for transferring digital value [3]. Further, Buterin (2014) extended this concept with Ethereum, introducing programmable smart contracts to automate transactions and decentralized finance (DeFi) applications [4]. On the interoperability front, Kshetri (2018) highlighted the need for bridging traditional financial systems with blockchain to unlock new efficiencies, especially in cross-border remittances and micropayments [5]. Gans (2019) emphasized that consumer-facing crypto applications must abstract the complexity of blockchain to gain mainstream appeal [6]. Recent developments have also focused on hybrid payment systems. For instance, Chen et al. (2022) proposed a dual-layer wallet system combining fiat and crypto functionalities, showcasing its viability for retail use cases [7]. However, these solutions often lack full integration or regulatory alignment in the Indian context. This literature confirms the need for a unified, user-centric system like UniPay, which leverages the strengths of both ecosystems while addressing their limitations.

III. PROBLEM STATEMENT

Currently, there is no robust infrastructure that allows seamless, real-time interoperability between fiat and cryptocurrency in a user-friendly interface. Users are forced to rely on:

1) Multiple applications for crypto wallets and fiat banking,
2) Long, complex public key addresses for crypto transfers,
3) Volatile exchange rate risks during conversions,
4) Unclear regulatory frameworks for compliant cross-system transactions.

This fragmentation limits adoption among consumers and merchants alike. There is an urgent need for a unified platform that simplifies this experience without compromising on security, scalability, and compliance.

IV. OBJECTIVES

The objectives of the UniPay platform include:

1) Unification of fiat and crypto payments through a common gateway.
2) Instant conversion between INR and cryptocurrencies using live market rates.
3) Integration with UPI for fiat and Ethereum-compatible wallets for crypto.
4) User identity abstraction, replacing public key usage with human-readable handles.
5) Compliance and security via KYC/AML integration and encrypted transaction flows.

V. METHODOLOGY

The development of UniPay involves a modular architecture that enables interaction between centralized banking systems and decentralized blockchain networks. The core of the system includes four major components:

Unified Wallet Interface: This is the user-facing layer that consolidates both fiat and cryptocurrency balances into a single view. Users can send, receive, and manage their funds using a clean, intuitive interface. The wallet supports bank account linking via UPI for fiat transactions and private-public key wallets for crypto.

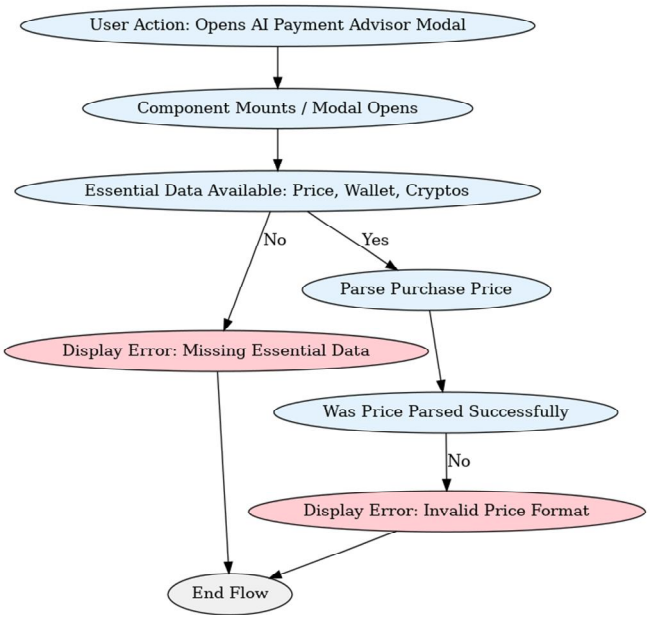
Transaction Routing Engine: The platform includes a transaction engine that intelligently determines whether the payment is fiat or crypto, and routes it to the appropriate backend system. If conversion is required, the routing engine interfaces with a liquidity provider or decentralized exchange (DEX) to execute the trade in real-time.

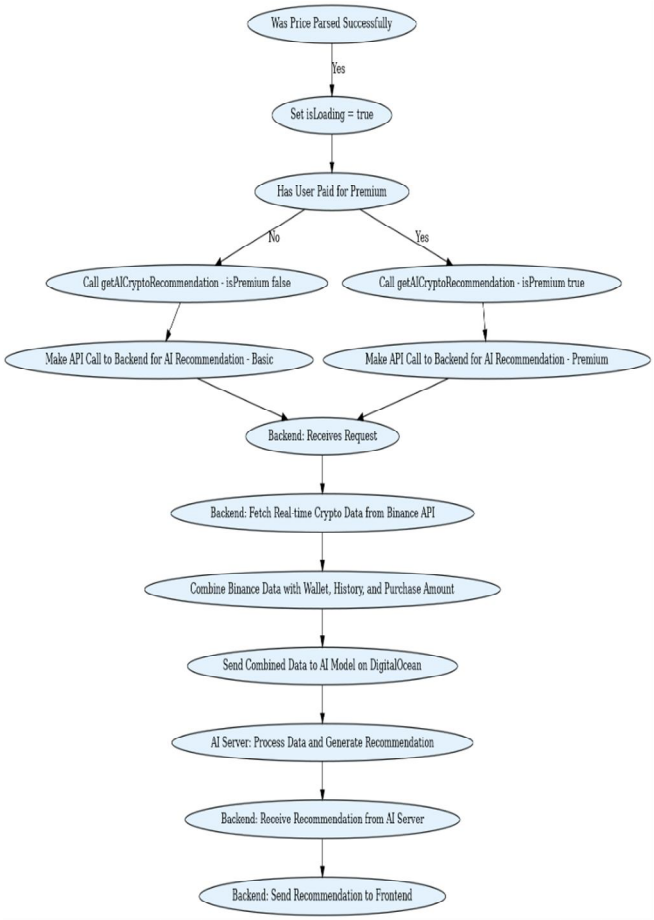
Blockchain Integration Layer: For cryptocurrency transactions, UniPay utilizes APIs and smart contract mechanisms built on Ethereum (and later extended to other chains like Solana, Polygon, etc.). These smart contracts handle payment verification, identity mapping, and token transfers.

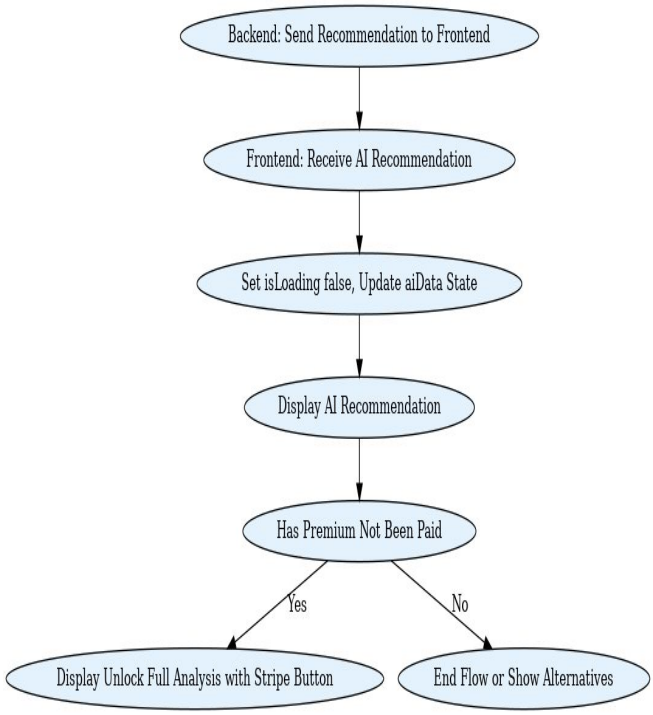
Banking API Gateway: This component securely communicates with banking partners, UPI APIs, and the NPCI framework to facilitate fiat transactions. We use OAuth2.0 for secure authorization and implement AES encryption to protect user data during banking operations.

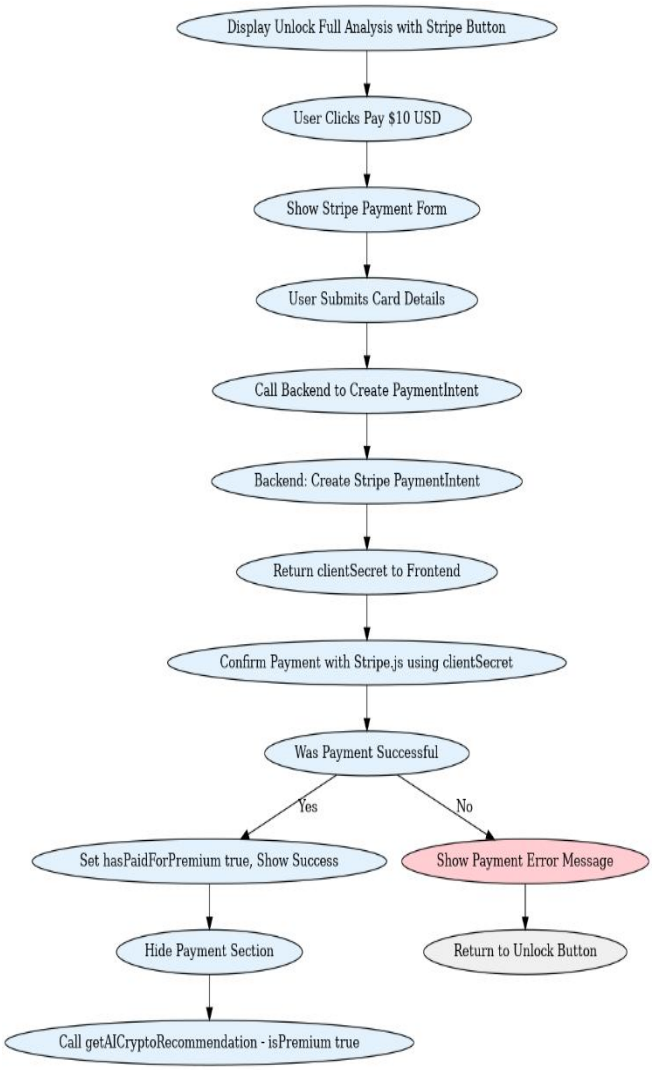
The platform also features robust identity verification mechanisms. During onboarding, users must complete KYC verification, which binds their identity to a universal transaction ID. This ID can be used to send or receive payments using human-readable handles (e.g., @username), reducing the need to remember long wallet addresses or account numbers. Additionally, the platform uses real-time AML (Anti-Money Laundering) screening and anomaly detection algorithms powered by machine learning to flag suspicious transactions and ensure compliance with financial regulations.

VI. SYSTEM ARCHITECTURE




Was Price Parsed Successfully
Yes
Set isLoading = true
Has User Paid for Premium
No
Yes
Call getAICryptoRecommendation - isPremium false
Call getAICryptoRecommendation - isPremium true
Make API Call to Backend for AI Recommendation - Basic
Make API Call to Backend for AI Recommendation - Premium
Backend: Receives Request
Backend: Fetch Real-time Crypto Data from Binance API
Combine Binance Data with Wallet, History, and Purchase Amount
Send Combined Data to AI Model on DigitalOcean
AI Server: Process Data and Generate Recommendation
Backend: Receive Recommendation from AI Server
Backend: Send Recommendation to Frontend


Backend: Send Recommendation to Frontend
Frontend: Receive AI Recommendation
Set isLoading false, Update aiData State
Display AI Recommendation
Has Premium Not Been Paid
Yes
No
Display Unlock Full Analysis with Stripe Button
End Flow or Show Alternatives



VII. IMPLEMENTATION

Our implementation focused on building a blockchain-based platform using Ethereum smart contracts for secure, verifiable, and tamper-proof record storage and transaction management. The objective was to utilize the core strengths of blockchain—immutability, decentralization, and transparency—to address challenges related to central authority dependence and trust-based systems.

A. Platform Selection: Ethereum

We opted to build the solution on Ethereum due to its:

1) Well-established infrastructure and tooling.
2) Large and active developer community.
3) Robust support for Turing-complete smart contracts through Solidity.
4) Seamless integration with browser-based wallets like MetaMask.
5) Reasonable block time (~15 seconds) that allows for relatively fast updates on the blockchain.
6) The smart contracts were written in Solidity and deployed using Truffle and Ganache for local testing before migration to the Goerli test network.

B. Architecture Overview

1) The system is composed of the following components:
2) Smart Contracts (Solidity)
3) Frontend Interface (ReactJS + Next.js)
4) Web3 Integration
5) Wallet Interface (MetaMask)

The application operates entirely on decentralized logic, with smart contracts handling core operations, while the frontend facilitates user interaction and transaction initiation.

C. Smart Contract Design

The smart contract layer is the core of our decentralized application (DApp). The contracts were designed to handle the following operations:

1) Registering Users: Each participant in the system—such as an issuer, verifier, or recipient—is associated with a unique Ethereum address. Role-based access was implemented using Solidity modifiers.
2) Creating Records: Authorized issuers can create new records on the blockchain. Each record contains structured data and is assigned a unique identifier (e.g., a hash of metadata), ensuring immutability.
3) Verification: Any participant can verify the integrity and authenticity of a record by querying the smart contract. Since all entries are stored on-chain, this enables zero-trust, decentralized verification.
4) Auditability: Every change or interaction with a record is automatically time-stamped and logged on-chain, enabling a transparent audit trail without a central authority.

The contracts were designed modularly to ensure that they can be extended in the future—for example, adding functionality such as record expiration, revocation, or layered authorization.

D. Frontend and User Interface

The user interface was developed using ReactJS and Next.js, providing a fast, responsive, and intuitive experience. Key features included:

1) Role-based Access Panels: Different dashboards were available for users based on their roles—issuer, verifier, or general user.
2) Transaction Workflow Interface: Users could fill out a form to initiate a blockchain transaction. For example, to register a new certificate, the issuer would fill in relevant fields, and upon clicking “Submit,” the transaction would be signed and sent via MetaMask.
3) Verification Portal: A dedicated verification page allowed users to input a record hash or ID and query the blockchain for authenticity.
4) Record History Viewer: Enabled browsing all entries created by a particular issuer or associated with a specific address.

All blockchain interactions were handled through Web3.js, which connected the frontend to the smart contracts deployed on Ethereum.

E. MetaMask Integration

MetaMask was used to handle:

1) User authentication (via wallet address).
2) Transaction signing (secure and non-custodial).
3) Network selection (connecting to Goerli test net for testing).
4) By integrating MetaMask, we avoided the need to build a separate login system, instead leveraging blockchain-based identity.

F. Development Stack and Tools

1) Solidity for writing smart contracts.
2) Truffle Suite for development, testing, and deployment.
3) Ganache for local blockchain simulation.
4) ReactJS + Next.js for the frontend.
5) Web3.js for blockchain integration.
6) Open Zeppelin contracts for secure standard implementations (like ownership and access control).

G. Sample Workflow

Here's an example of a typical user journey:

1) Issuer logs in using MetaMask and navigates to the "Create Record" page.
2) They enter record details (e.g., student name, ID, certificate hash) and submit.
3) The createRecord() function is called in the smart contract, and the transaction is sent to the Ethereum network.
4) Once mined, the transaction hash and block ID are shown to the user, confirming the successful registration.
5) Later, a verifier enters the record ID in the "Verify Record" page.
6) The system queries the smart contract and validates the record's authenticity and issuer address.

This implementation demonstrates the potential of smart contracts and blockchain in building transparent, verifiable, and secure systems without reliance on centralized control. The modular architecture also ensures scalability for future use cases—such as education certificates, digital identities, or document verification systems.

VIII.RESULTS AND DISCUSSIONS

The prototype implementation of our blockchain-based record verification system was successfully deployed and tested on the Ethereum Goerli test network. The smart contracts demonstrated secure, immutable storage of records and efficient retrieval for verification purposes. The platform exhibited the following key outcomes:

1) Immutable Record Creation: All records stored on the blockchain remained tamper-proof and auditable. Attempts to overwrite or delete existing records were correctly rejected by the smart contract's internal logic.
2) Real-time Verification: Verifiers could instantly confirm the authenticity of any record using the unique identifier (hash), showcasing the real-time transparency advantage of blockchain.
3) Role-Based Access: The smart contract enforced strict access control, ensuring only authorized users (e.g., issuers) could create records. Unauthorized transaction attempts failed as expected, highlighting secure permission control.
4) Minimal Latency: With the average Ethereum block time of ~15 seconds, transactions were confirmed within an acceptable delay, validating the platform's practicality for near real-time applications.
5) Gas Optimization: Smart contract functions were optimized for minimal gas usage by reducing redundant state changes and leveraging mapping structures. This made the system relatively cost-effective to use on Ethereum testnets and suggested feasibility on mainnet with further optimization.
6) User Interface Performance: The frontend built using ReactJS and Web3.js provided a smooth experience with responsive feedback from MetaMask and contract interactions, making it easy even for non-technical users to access blockchain features.

However, challenges were encountered with:

a) Occasional MetaMask disconnections requiring manual refresh or re-login.
b) Network congestion delays during high testnet traffic times.
c) The lack of off-chain storage (for larger metadata) required clever hashing and referencing techniques.

Overall, the results validate the project's core promise: a secure, decentralized, and transparent platform for record issuance and verification..

IX. FUTURE SCOPE

T While the current implementation focuses on secure on-chain record verification, there are several avenues for future expansion:

1) Integration with UPI for Payment-Linked Services:

A future version of the platform can incorporate Unified Payments Interface (UPI) to facilitate:

- Payment processing for certificate issuance fees.
- Transaction-based verification access (e.g., charging users a small amount to verify high-value records).
- Issuer payouts or subscription-based billing for record storage.

Integration can be implemented using UPI-enabled payment gateways and connecting them through APIs to a Node.js backend, which in turn links the transaction metadata with the blockchain transaction.

2) Decentralized File Storage (e.g., IPFS): Storing large documents (e.g., degree certificates, PDFs) on-chain is costly. The InterPlanetary File System (IPFS) can be used to store files, and only the content hash would be stored on Ethereum.
3) Support for Multiple Blockchain Networks: Future versions could provide interoperability with other EVM-compatible chains like Polygon, Binance Smart Chain, or even non-EVM chains like Solana for scalability and lower fees.

4) Decentralized Identity (DID) Integration: Adding support for DIDs and Self-Sovereign Identity (SSI) protocols to authenticate users without centralized logins.

5) Mobile App Development: A React Native mobile app could increase accessibility, especially for field-based use cases like on-site verification.

X. CONCLUSION

This project successfully demonstrates the development of a decentralized, Ethereum-based platform for secure and transparent record management. By harnessing smart contracts, the system eliminates the need for intermediaries, ensures data immutability, and enables real-time verification. The implementation validates key blockchain principles—decentralization, trustlessness, and auditability—in a real-world application setting.

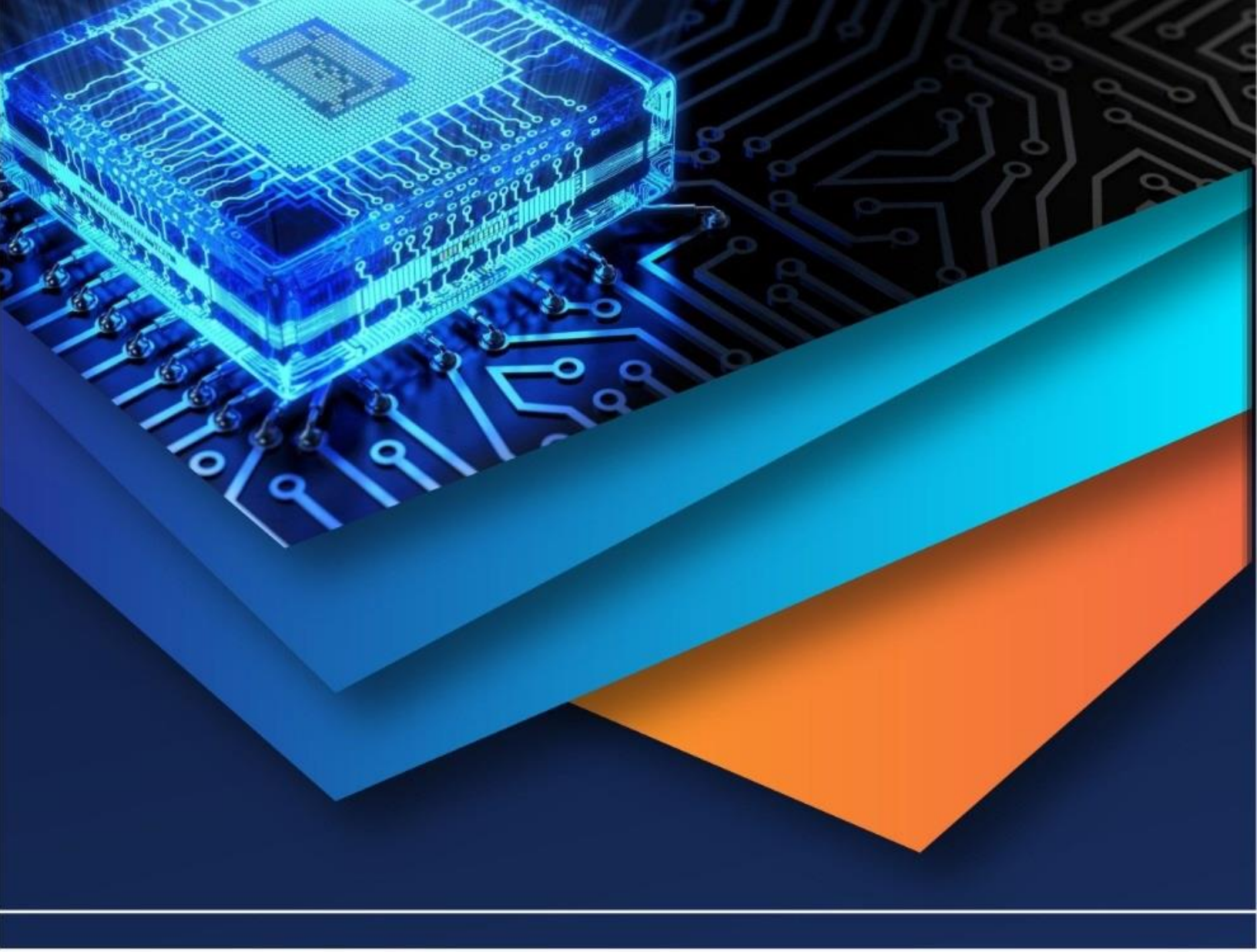
The prototype provides a scalable base architecture for integrating more advanced features like UPI payments, decentralized file storage, and mobile access. As blockchain adoption grows and regulatory frameworks mature, systems like ours can play a vital role in transforming trust-based systems into decentralized ecosystems with provable integrity and minimal overhead.

XI. ACKNOWLEDGMENT

Special thanks to Vishwakarma Institute of Technology for providing us various resources and guidance.

REFERENCES

[1] Sarkar, S., & Dey, R. (2021). UPI and Financial Inclusion: A Study on the Impact of UPI Transactions in India. International Journal of Research in Business and Social Science, 10(2), 143–150.
[2] Gupta, V., & Jain, A. (2020). Security Architecture of UPI: A Case Study of Indian Digital Payments. Journal of Cybersecurity and Information Management, 8(1), 35–41.
[3] Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Whitepaper.
[4] Buterin, V. (2014). A Next-Generation Smart Contract and Decentralized Application Platform. Ethereum White Paper.
[5] Kshetri, N. (2018). 1 Blockchain's roles in meeting key supply chain management objectives. International Journal of Information Management, 39, 80–89
[6] Gans, J. S. (2019). The Case for an Interoperable Payments Infrastructure. NBER Working Paper No. 26021.
[7] Chen, W., Lin, L., & Zhang, M. (2022). Design of a Hybrid Payment System: Integrating Centralized Fiat and Decentralized Crypto Wallets. Journal of Financial Innovation and Technology, 3(2), 112–126.
[8] Zyskind, G., Nathan, O., & Pentland, A. (2015). Decentralizing privacy: Using blockchain to protect personal data. 2015 IEEE Security and Privacy Workshops, 180–184.
[9] Christidis, K., & Devetsikiotis, M. (2016). Blockchains and smart contracts for the Internet of Things. IEEE Access, 4, 2292–2303.
[10] Al-Bassam, M. (2017). Scalable secure multi-party computation using blockchain. arXiv preprint arXiv:1707.05491.
[11] Saberi, S., Kouhizadeh, M., Sarkis, J., & Shen, L. (2019). Blockchain technology and its relationships to sustainable supply chain management. International Journal of Production Research, 57(7), 2117–2135.
[12] Narayanan, A., Bonneau, J., Felten, E., Miller, A., & Goldfeder, S. (2016). Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction. Princeton University Press.
[13] Jain, A., & Bansal, V. (2022). UPI-Based Micro-Payments and Blockchain: A Comparative Review for Secure Transactions in India. International Journal of Financial Technologies and Digital Innovation, 1(1), 55–68.
[14] Szmigiera, M. (2022). Adoption of UPI in India: A Statistical Overview. Statista Research Department.
[15] Li, X., Jiang, P., Chen, T., Luo, X., & Wen, Q. (2020). A survey on the security of blockchain systems. Future Generation Computer Systems, 107, 841–853.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY