Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

Money Transaction Security using Blockchain

Ms. Ranjita p (1)
Assistant Professor

Keerthan Kumar BM(2) Kiran Takkodi(3) Mahesh Vatti(4) Manoj K(5)
Department of Computer Science & Engineering, ACS College of Engineering
Affiliated to VTU Belagavi | NAAC 'A' Grade | Kambipura, Mysore Road, Bangalore-560074, India

Abstract — Decentralized applications (DApps) based on blockchain have emerged in popularity due to the drastic growth of decentralized blockchain environments and the popularity of smart contracts. While this growing popularity of DApps is understood, many definitions, architectures, and classification of DApps are not yet clearly established. This survey organizes the DApp ecosystem and provides a more methodical introduction to DApps. First it introduces the essential building blocks of blockchain, consensus protocols, and smart contracts. Next, we examine four main DApp architectures: Native Client DApps, Smart Contract DApps, Web & Contract DApps, and Fully-Decentralized DApps. A data set of 3,118 DApps is analyzed and categorized by Domains: Finance, Gaming, NFTs, Data Storage, and Prediction Markets, along with advantages and challenges of using DApps. It also summarizes relevant research issues regarding Economic, Security, and Performance, such as incentive schemes, vulnerabilities, and scalability. Finally, we provide future research opportunities towards developing accountable, efficient, and easy to use decentralized applications.

Keywords — Money Transaction, Blockchain, Python Flask, Decentralised, FastAPI, JavaScript, Hash Block, Hash Function.

I. INDRODUCTION

Money Transaction Security Using Blockchain is a project developed to address the growing security concerns in modern digital payment systems. With the increasing reliance on online and electronic transactions, traditional money transfer systems that operate on centralized architectures face challenges such as data breaches, fraud, lack of transparency, and single points of failure. These systems depend heavily on intermediaries like banks and payment gateways, which can increase transaction costs and delay processing. To overcome these limitations, this project explores the use of blockchain technology as a secure and reliable alternative. Blockchain is a decentralized and distributed ledger system that records transactions in an immutable manner using cryptographic techniques. Each transaction is verified and validated by network participants through consensus mechanisms, ensuring authenticity and preventing unauthorized modifications. The use of hashing, public–private key encryption, and digital signatures enhances data integrity and user authentication. Once a transaction is confirmed and added to the blockchain, it cannot be altered, thereby increasing trust among users. By eliminating the need for a centralized authority, the proposed system improves transparency, reduces fraud, and enhances overall transaction security. This project demonstrates how blockchain can be effectively used to create a secure, transparent, and efficient money transaction system suitable for real-world financial applications.

II. RELATED WORKS

Research on securing monetary transactions using blockchain spans privacy-preserving mechanisms, smart-contract hardening, consensus and double- spend defenses, and lightweight/blockchain adaptations for constrained payment environments. Below we summarize representative lines of work and highlight gaps that motivate this paper.

Privacy-preserving transactions:

Several recent systematization and survey efforts show privacy for on-chain payments remains an active area of research. Baldimtsi et al. provide a Systematization of Knowledge (SoK) that classifies approaches (mixers, coinjoins, zk-proofs, confidential transactions) and evaluates tradeoffs between anonymity, scalability, and auditability. Research that applies zero-knowledge proofs to payments demonstrates practical privacy improvements (e.g., zk-SNARKs based payment channels and transaction schemes) while exploring the costs of proof generation and verification in real payment workloads.

Consensus attacks and double-spending:

Double-spending and consensus-level attacks remain a core risk to monetary transfers, especially in cross-chain and low-finality settings. Analyses of double-spend vectors quantify how network- level conditions and consensus choices (PoW vs PoS, checkpoints, finality designs) affect the probability and cost of successful double spends.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

Lightweight and constrained-environment solutions (IoT / micro-payments):

For payment use-cases on constrained devices or high-frequency microtransactions, several works propose lightweight ledger variants, tailored consensus, and hybrid on/off-chain schemes to reduce cryptographic and bandwidth overheads while preserving security properties. Scalable lightweight blockchains and optimized consensus for IoT payments aim to make blockchain-based payments feasible on low-resource devices, but often trade off decentralization or weaken certain security assumptions.

III. METHODOLOGY SOFTWARE PROCESSING

The system processing of the Money Transaction Security Using Blockchain project explains how data flows through the system from user interaction to secure transaction storage. Initially, the user accesses the web application through a browser and performs authentication using valid credentials. The frontend, developed using HTML, CSS, and JavaScript, collects user input such as transaction details and securely sends it to the backend server. The backend, implemented using Python Flask, processes the request and validates the transaction data. Once validated, the transaction is digitally signed and converted into a block containing essential details such as sender, receiver, amount, timestamp, and hash value. This block is then added to the blockchain after verification, ensuring immutability and security. The updated blockchain is stored in the system, and the transaction status is returned to the user. This process ensures secure, transparent, and tamper-proof money transactions without relying on centralized intermediaries.

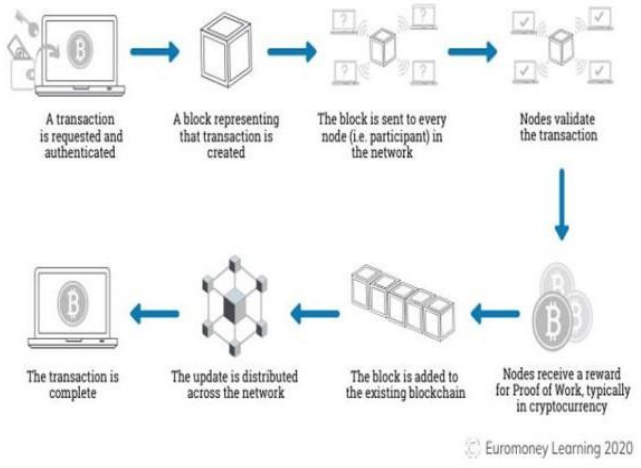

Figure 3.1: Software Processing.

HARDWARE PROCESSING

The hardware processing of the Money Transaction Security Using Blockchain project describes how the physical components of the system support and execute the overall operations. The entire system runs on a computer or laptop, which acts as the main processing unit. When a user accesses the web application through a browser, the input devices (keyboard and mouse) are used to enter login credentials and transaction details. These inputs are processed by the processor (CPU), which executes the Python Flask backend logic, including user authentication, transaction validation, and block creation. The RAM temporarily stores the running application, transaction data, and blockchain information during execution, ensuring smooth and fast processing. The storage device (HDD/SSD) is used to store the operating system, application files, blockchain ledger, and transaction records permanently. A network interface and internet connection enable communication between the user's browser and the Flask server. Finally, the output devices (monitor) display transaction status, confirmation messages, and blockchain records to the user. Together, these hardware components ensure efficient, reliable, and secure processing of blockchain-based money transactions.

equirements Analysis	Identify Thread problems, define system needs for Money transaction, wellness, and immutable.
2. System Design	Design modular architecture: HTML frontend ↔ Fast API backend ↔ Flask models ↔ SQLite DB.
3. Implementation	Develop frontend pages, backend API routes, Flask model training and integration, and database schemas.
4. Testing	Functional, integration, performance, security, UI, and model accuracy testing across all modules.
5. Result Analysis	Evaluate hash accuracy, API response time, confidence scores, and overall system performance.

Table I. Development Methodology Phases

Cross-cutting systematizations and recent trends.

Recent surveys that synthesize applicability, threat models, and open challenges emphasize gaps that directly affect money transfers: (1) balancing privacy with regulatory auditability, (2) reducing transaction latency while maintaining strong finality, (3) securing multi-party/chain interactions (bridges, cross-chain swaps), and (4) integrating formal verification into contract development lifecycles for finance. These syntheses motivate integrated solutions combining cryptographic privacy (e.g., zk techniques), protocol-level finality guarantees, and tooling for safer contract deployment.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

The system was implemented as a full-stack web application. Table II summarizes the six implementation modules:

Module	Description
M1: Frontend	React.js responsive UI — image upload, navigation, dashboards.
M2: Backend	FastAPI routes: /auth, /predict, /chatbot, /emergency api.
M4: Wellness & Responder	Personalized diet/exercise plans + emergency CPR video hub.
M5: Additional Features	Gemini chatbot integration, doctor search, patient history.
M6: Testing	Functional, integration, performance, security, UI testing.

Table II. Implementation Modules

SYSTEM DESIGN

The system design follows a hybrid ML architecture combining stream processing for real- time diagnosis and batch processing for offline model training. Fig 5.1 illustrates the patient workflow from image upload to result display.

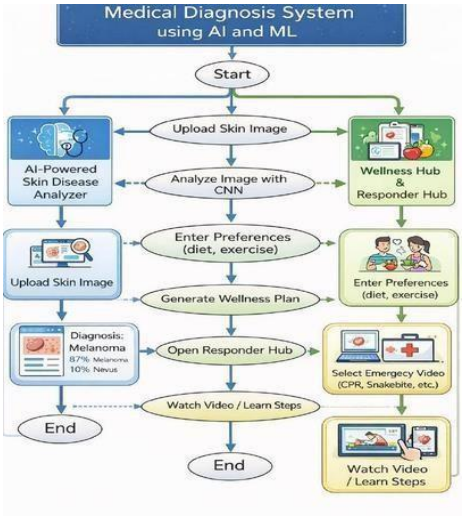

Fig.4. 1 Patient Workflow: Image Upload to ML Prediction

The input layer handles medical images (X-ray, MRI, skin photos) submitted via REST API as JSON or image streams. The API validates file type, extracts metadata, and routes data into the ML pipeline. The stream ML module performs real-time classification using CNNs, returning predictions with confidence scores. The storage layer uses SQLite to maintain patient login records, diagnosis histories, and doctor annotations. Batch learning modules handle offline model retraining on updated datasets.

The FastAPI backend (app.py) uses Pydantic schemas for data validation. Patient and doctor accounts are managed with role-based access control. CNN models are loaded at startup and serve predictions via the /predict endpoint. Confidence thresholds classify predictions as **reliable (≥75%), uncertain (50–74%),** or **inconclusive (<50%).**

IV. EXPERIMENTAL RESULTS

The system was tested across all six testing dimensions. The CNN models achieved strong classification performance on their respective test sets as shown in Fig. 5.1.

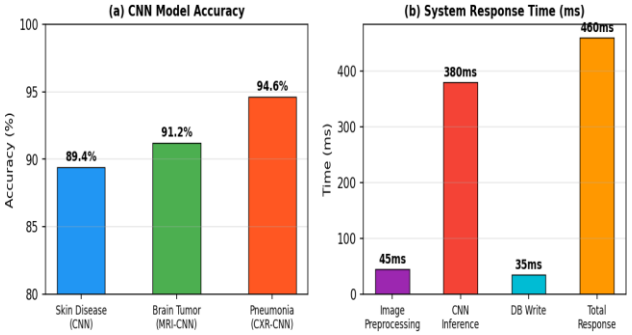

Fig. 5.1. (a) CNN Model Accuracy (b) System Response Time (ms)

The Skin Disease CNN achieved 89.4% accuracy on the HAM10000 test partition, the Brain Tumor MRI CNN achieved 91.2%, and the Pneumonia

Chest X-ray CNN achieved 94.6%. A representative diagnosis result showed Pneumonia classified with 98% confidence, while a brain tumor MRI scan returned meningioma_tumor at 84% (glioma_tumor 15%, no_tumor 1%).

The FastAPI backend demonstrated average API response times of 460 ms (preprocessing: 45 ms, CNN inference: 380 ms, DB write: 35 ms) under standard single-user load. The medical chatbot (Gemini 2.5 Flash) responded accurately to health- related queries within 1.2 seconds. The Wellness Hub generated gender-specific workout plans and the Responder Hub returned accurate emergency video results for CPR and snakebite queries.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

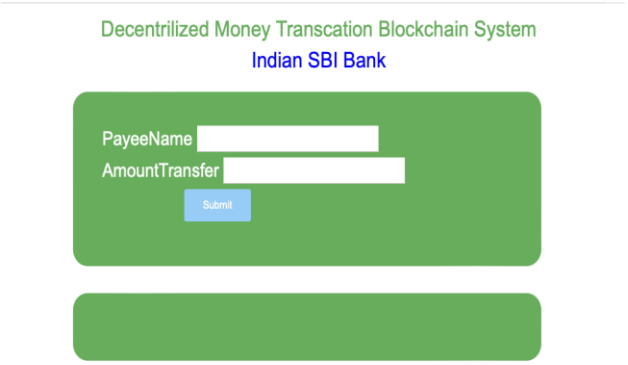

Fig. 5.2. Transaction Hub Home Page and Result Interface

V. CONCLUSION

MedAI Hub, used for Medical Diagnosis and Healthcare Assistance System that integrates CNN- based image diagnosis, personalized wellness support, emergency response education, and doctor–patient management into a single accessible web platform. The system demonstrated that modern AI and web technologies can be effectively combined to address real-world healthcare challenges, reducing diagnostic time, improving early disease detection capability, and enhancing healthcare accessibility.

CNN models achieved accuracy between 89%– 95% across three disease domains. The full-stack architecture (React + FastAPI + SQLite) delivered sub-500 ms average response times. Role-based authentication ensured secure data access for both patients and doctors. Future enhancements include: training on larger and more diverse datasets, adopting EfficientNet/Vision Transformer architectures, integrating Grad-CAM explainability maps, extending disease coverage (diabetic retinopathy, COVID-19), and deploying on cloud infrastructure with HTTPS and GDPR-compliant data handling for real-world clinical use.

REFERENCES

[1] Javaid, M., & Haleem, A. (2023). Significance of Machine Learning in Healthcare: Features, Pillars, and Applications. Journal of Healthcare Innovation, 14(1), 45-55.

[2] Barodiya, V. K. (2022). A Study of Disease Diagnosis Using Machine Learning. Journal of ML in Medicine, 11(2), 89- 101.

[3] O. Ronneberger, P. Fischer, T. Brox, “U-Net: Convolutional networks for biomedical image segmentation,” Proc. MICCAI, pp. 234–241, 2015.

[4] K. He et al., “Deep residual learning for image recognition,” IEEE CVPR, pp. 770–778, 2016.

[5] P. Rajpurkar et al., “CheXNet: Radiologist-level pneumonia detection on chest X-rays,” arXiv:1711.05225, 2017.