e-ISSN : 2541-2019
p-ISSN : 2541-044X

Blockchain Disaster-Relief DApps with SVM and Data Anchors for Fraud-Prevention

Agil Zaky Ardhi[1)], **Ratih Titi Komala Sari**[2)*], **Novi Dian Nathasia**[3)], **Sari Ningsih**[4)]
[1,2,3,4)]Informatika, Fakultas Teknologi Komunikasi dan Informatika, Universitas Nasional
[1)]22agil.zaky@civitas.unas.ac.id, [2)]ratih@civitas.unas.ac.id, [3)]novidian@civitas.unas.ac.id
[4)]sari.ningsih@civitas.unas.ac.id

Submitted :Oct 31, 2025 | **Accepted** : Nov 28, 2025 | **Published** : Jan 02, 2026

Abstract: VoucherAid and DataAnchor are prototype DApps for disaster-relief voucher processing that integrate on-chain rule enforcement, cryptographic data anchoring through fixed-size hash commitments, and an off-chain SVM-based analytics gateway. VoucherAid issues non-transferable vouchers, restricts redemption to certified merchants, and emits auditable events, while DataAnchor records time-stamped digests to support provenance verification without exposing sensitive content. A 200-record dataset was generated from on-chain logs and enriched with behavioral–temporal features derived from redemption activity. Experiments conducted in a single-node Ganache environment using a 70:30 split show that the SVM achieves 0.75 accuracy with perfect precision but limited recall for fraud (1.00 precision, 0.32 recall, 0.48 F1), indicating that the model cannot serve as a reliable stand-alone detector and is more appropriate as a conservative decision-support tool under human oversight. The prototype demonstrates that separating on-chain enforcement from off-chain analytics can enhance auditability and support model evolution without contract redeployment. However, the findings remain constrained by the small, partially synthetic dataset, the single-node evaluation environment, and programmatic labeling. Future work will expand datasets, incorporate richer temporal and graph-based features, adjust thresholds and class weights, and evaluate the system on multi-node networks to improve fraud recall while maintaining usability and inclusion.
Keywords: Blockchain; Disaster Relief; Digital Vouchers; Fraud Detection; Support Vector Machine.

INTRODUCTION

Disaster response operations often struggle with transparency in aid distribution and the integrity of situational data for decision-making. Centralized systems are fast but become opaque when many stakeholders are involved, creating risks of fraud such as merchant collusion, burst redemption, and data disputes. Blockchain approaches promise end-to-end traceability, yet many prototypes stop at basic token transfers without fraud-prevention layers or protection for upstream data (Kaur & Singh, 2023; Rejeb et al., 2022). This paper introduces VoucherAid and DataAnchor, minimalist DApps that combine smart contract–based voucher distribution with cryptographic anchoring (e.g., Merkle roots) of early-warning and operational data. VoucherAid issues non-transferable vouchers, restricts redemption to certified merchants, and emits auditable events, while DataAnchor records timestamped hash commitments on-chain to enable public verification without placing sensitive data on-chain.

To proactively mitigate abuse, we integrate a supervised Support Vector Machine (SVM) (Hindarto & Santoso, 2022) classifier that scores voucher redemption attempts using behavioral and temporal features derived from on-chain logs Li et al., (2022) and auxiliary metadata. Example features include redemption frequency by beneficiary and merchant, inter-arrival gaps, merchant switching patterns over a 24-hour window, and beneficiary-specific z-scores for amount size. The SVM runs off-chain inside a gateway that mediates user actions before transaction submission. High-risk attempts are blocked or flagged at the edge, preserving gas, reducing on-chain clutter, and preventing clearly suspicious transactions from ever reaching the contract. This design keeps the core ledger simple, verifiable, and inexpensive, while allowing the fraud logic to evolve without redeploying contracts.

The prototype targets an Ethereum-compatible environment using Ganache/Truffle in a single-node simulation (Hindarto, Rachmadi, et al., 2025). Although simplified, this setup offers three advantages that matter in early research: controlled determinism for reproducible results, low operational friction that enables rapid iteration on contract and feature engineering, and the ability to inject synthetic attack scenarios at scale. The SVM is trained using stratified cross-validation with class-balancing to reflect the natural rarity of fraud. We report

*name of corresponding author

metrics that matter for operations—fraud recall, fraud F1, and precision–recall trade-offs—rather than only accuracy, which can be misleading under class imbalance.

The study has three objectives. First, we design and implement a verifiable voucher workflow with merchant whitelisting, event emission, and clear administrative controls (e.g., pausing and blocking) (Hindarto, Damastuti, et al., 2025). Second, we secure the integrity and provenance of early-warning and operational data through on-chain anchoring of cryptographic commitments, so investigators can later verify that decisions were driven by unaltered inputs. Third, we evaluate an SVM-based fraud-prevention module for real-time screening of suspicious redemptions, studying its decision thresholds and operational implications (false positives that delay legitimate aid versus false negatives that permit abuse).

These objectives translate into the following research questions: RQ1. How well can a lightweight, single node blockchain prototype provide transparent audit trails without harming usability for frontline actors? RQ2. Does on-chain anchoring materially improve the verifiability of upstream sensor/operational data that informs relief workflows? RQ3. Under realistic constraints (small, imbalanced data and synthetic labels), what precision–recall trade-offs can an SVM classifier achieve when screening voucher redemptions for suspicious patterns, and is its performance sufficient for use as an operational decision aid?

Our contributions are threefold. We deliver an open, minimal DApps architecture that separates concerns: contracts enforce rules and emit events; off-chain services handle identity, data storage, and analytics; an AI module screens transactions before they hit the chain. We provide a concrete feature schema and training procedure for SVM-based fraud detection tailored to voucher redemption logs, along with a discussion of model explainability in an operational context. Finally, we present a reproducible simulation that blends contract execution, data anchoring, and AI screening, designed to be extended toward multi-node testnets or integrated with real sensor feeds.

Admittedly, this work does not claim to solve the sociotechnical complexity of disaster relief. A single-node simulation cannot capture network partitions, heterogeneous device failures, or the human factors that shape compliance. The SVM relies on labeled data, and labels produced from synthetic scenarios may not fully reflect real-world ingenuity. Nonetheless, a carefully instrumented prototype lets us test the spine of the system—verifiable workflows, auditable events, integrity guarantees, and risk-aware gating—before investing in heavier infrastructure or live deployments. The broader research program is to evolve from reproducible single-node experiments to field-ready, privacy-respecting systems that coordinate multiple institutions without collapsing under their own complexity.

LITERATURE REVIEW

Research on blockchain for humanitarian operations emphasizes transparency, traceability, and audit trails across fragmented networks of actors (governments, NGOs, merchants, recipients) (Zhou et al., 2024). Studies on aid distribution and voucher schemes report lower reconciliation costs and clearer audit trails when approvals/transfers are encoded in smart contracts rather than spreadsheets or siloed databases. Humanitarian supply chain research extends this logic to goods tracking—recording ownership transfers and delivery confirmations to reduce leakage and data manipulation (Rejeb et al., 2022). However, many prototypes stop at asset transfer logic (mint/burn tokens) and do not yet incorporate decision triggers from upstream data (e.g., early warnings), limiting their impact to operations that are highly dependent on timing (Hunt et al., 2022).

The complementary literature discusses securing data provenance through anchoring—periodically committing cryptographic digests (e.g., Merkle roots) of off-chain payloads to a public ledger (Li et al., 2022). Anchoring provides an immutable timestamp and tamper-proof evidence without publishing sensitive content, useful for sensor networks (hydrology, seismology) and operational logs. Oracle frameworks bridge off-chain measurements to on-chain decisions, but challenges exist regarding trust, latency, and potential censorship (Sharma et al., 2023); suggested solutions include multi-publisher (N-of-M), weighted updates, and verifiable aggregation. While promising, few disaster response implementations evaluate anchoring latency/jitter under stress or demonstrate how anchored data improves post-disaster auditing (Patel et al., 2024).

Work on digital identity—specifically self-sovereign identity (SSI), decentralized identifiers (DID), and verifiable credentials (VC)—aims to express eligibility/role/program rules while minimizing data exposure (Kaur & Singh, 2023). Voucher systems benefit from credential-based access control (e.g., “beneficiary verified,” “merchant certified”) with selective disclosure to maintain privacy. However, operational studies highlight key recovery friction, low connectivity, and device loss in crisis contexts; production designs therefore combine custodial wallets, SMS/USSD fallbacks, and off-chain KYC with on-chain commitments. Rigorous evidence on the impact of these choices on inclusion/exclusion error and fairness remains limited (Feulner et al., 2025).

The literature on financial/e-commerce crime provides a rich set of tools for detecting collusion, burst behavior, and account takeover. Classic supervised methods—Support Vector Machines (SVM), logistic regression, random forests—are powerful when feature engineering captures temporal patterns (arrival intervals, moving averages, z-scores) and relational structures (beneficiary–merchant graphs). SVM is competitive in high-dimensional spaces

*name of corresponding author

with limited samples due to margin maximization and kernelized boundaries. Unlabeled approaches (One-Class SVM, Isolation Forest, autoencoder) are useful when labels are scarce but are prone to threshold drift and concept drift; metrics emphasizing fraud recall are recommended. Although the results are promising, most studies focus on e-commerce/cards; domain-specific evidence on voucher redemption in disaster response—with low connectivity constraints and human oversight—remains scarce (Rtayli & Enneya, 2020).

Consistent system papers argue that analytical fraud and computationally intensive tasks should run off chain, while smart contracts focus on enforcing rules and emitting events. This separation maintains determinism, lowers gas costs, and allows analytical models to evolve without redeploying contracts. Event-driven pipelines (listen → build features → score → gate) are commonly used, as are risk-aware gateways that block/hold transactions based on model output before they enter the chain. However, there are dangers: feedback loops (attackers adapt), fairness (false positives are uneven), and governance (appeals, overrides, model decision auditability). Recommended practices include transparent logging and human-in-the-loop triage (Alghanmi et al., 2025; Sharma et al., 2023).

From the various threads above, a pattern emerges: blockchain excels at immutable coordination and audit trails; anchoring secures provenance; and SVM—as a classic ML—remains a strong baseline for fraud screening when features are well-designed. However, there are three gaps: (i) most humanitarian prototypes have not yet tied upstream data that is anchored (e.g., early warning signals) to downstream decisions in a measurable way; (ii) fraud model evaluations rarely use features derived directly from on-chain redemption events within real operational constraints; and (iii) few report end-to-end simulations that integrate contract execution, data anchoring, and AI-gating while maintaining privacy (on-chain commitments, off-chain payloads) (Hunt et al., 2022).

This study closes that gap by (i) implementing VoucherAid for redeeming non-transferable vouchers based on a whitelist with auditable events; (ii) implementing DataAnchor to commit windowed hashes of sensor/operational payloads so that integrity and timestamps can be publicly verified; and (iii) integrating a supervised SVM classifier that scores redemption attempts using behavioral/temporal features from on-chain logs and additional context, with decisions enforced at an off-chain gateway before transactions are submitted. Evaluations were conducted in a single-node Ganache/Truffle environment to produce controlled and reproducible tests of contract correctness, anchoring latency, and fraud filtering performance—as a foundation for multi-node testing and field trials with real sensor feeds and human-in-the-loop governance (Baharmand & Comes, 2019).

METHOD

This study proposes an end-to-end, verifiable pipeline for disaster-relief vouchers that combines smart-contract enforcement, cryptographic data anchoring, and AI-assisted fraud screening. The workflow is: (i) VoucherAid smart contract issues non-transferable vouchers and restricts redemption to whitelisted merchants while emitting auditable events; (ii) DataAnchor commits Merkle-root digests of early-warning/operational payloads to the chain to secure integrity and timestamps; (iii) an off-chain gateway listens to events, constructs features, scores each redemption attempt with a Support Vector Machine (SVM), and blocks/flags high-risk requests before submitting transactions. We implement a single-node Ethereum-compatible testbed (Ganache/Truffle) to allow controlled measurement of contract correctness, anchoring latency, and screening latency (Hassan et al., 2022), and we provide sequence diagrams and flowcharts (listen → feature build → SVM score → allow/hold) to clarify data/control flow.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 10, Issue 1, Januari 2026
DOI : https://doi.org/10.33395/sinkron.v10i1.15522
e-ISSN : 2541-2019
p-ISSN : 2541-044X

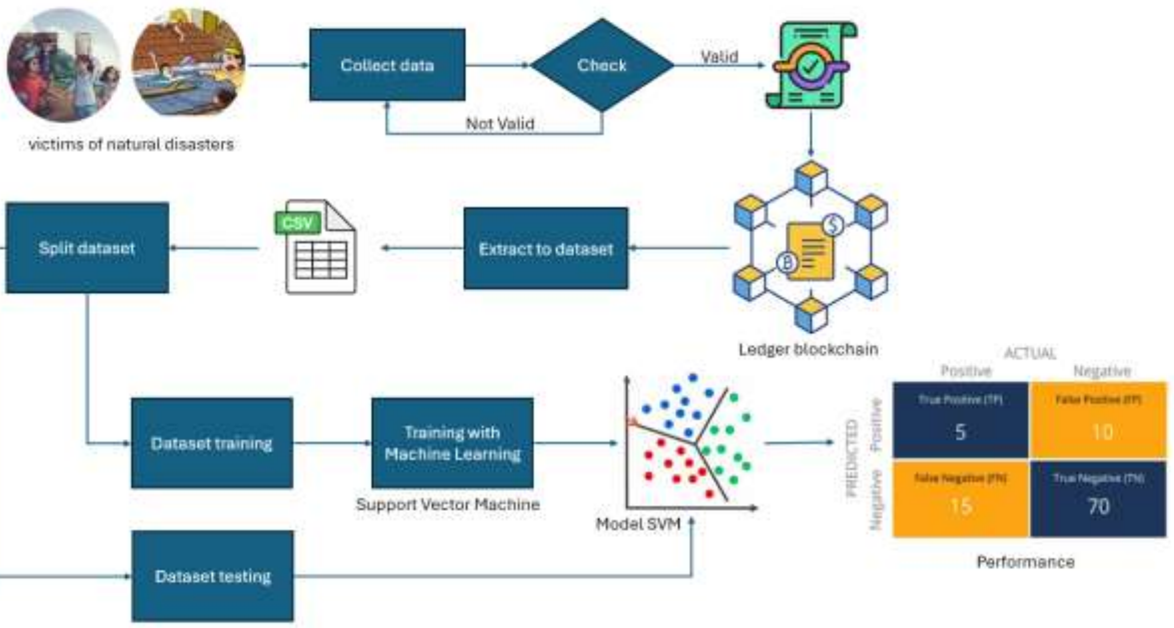


Fig. 1 Proposed method for a blockchain-based disaster victim donation system

Figure 1 details the methodological stages of a blockchain-enabled donation system for disaster victims. The workflow begins in the field—teams collect beneficiary data and supporting evidence—which then undergo validation to ensure identity completeness, verified disaster impact, and compliance with program criteria. Entries that pass validation are converted into cryptographic commitments and recorded on the blockchain ledger, making any modification detectable and each record time-stamped and tamper-evident. From these operational traces, transactions and redemption activities are extracted into a structured CSV dataset and subsequently partitioned into training and testing sets. During preprocessing, data are cleaned and standardized, then engineered into behavioral–temporal features—such as hourly redemption frequency, inter-arrival gaps, merchant switching by beneficiaries, 10-minute merchant crowding, and beneficiary-relative transaction magnitude (z-score). This design preserves a healthy separation of concerns: the blockchain guarantees integrity and traceability, while heavy computation and analytics run off chain to remain efficient and easily updatable.

The next stage is machine learning for fraud-risk mitigation. The engineered features are used to train a Support Vector Machine (SVM) as a binary classifier (legitimate vs. fraud) with an RBF kernel, allowing non-linear patterns—such as the combination of high merchant crowding, very short gaps, and above-normal amounts—to be modeled without complex architectures. The model is optimized via class balancing and decision-threshold selection based on precision–recall curves, with emphasis on fraud recall so high-risk cases are not missed. Performance is evaluated on the test set and summarized using a confusion matrix (TP, FP, FN, TN), which then informs the operating threshold and governance policies—e.g., holding transactions for manual review or retuning parameters as abuse patterns evolve. In this way, the design blends the immutability of blockchain with the agility of off-chain analytics to form a transparent, measurable, and field-adaptable pipeline for disaster response.

Dataset VoucherAid

We build a labeled dataset from on-chain events produced by VoucherAid during scripted simulations. Each row represents a redemption attempt with fields captured in Table 1: redemption frequency within the last 60-minute window (Freq), inter-arrival gap in seconds since the previous redemption by the same beneficiary (Arrival), merchant identifier (Merchant), standardized transaction amount based on beneficiary-level history (Amount), merchant crowding over a 10-minute window (Merchant crowd), synthetic beneficiary–merchant distance (Distance), and the binary label (Indicator). These engineered behavioral–temporal features capture burstiness, abnormal amounts, local crowding, and spatial inconsistencies that often characterize suspicious patterns. We omit hour-of-day and merchant-hopping features from the final table to maintain consistency with the implemented feature set. Labels are generated programmatically from scripted scenarios such as burst redemptions, collusive funnels to a single merchant, and extreme amounts. Importantly, the fraud labels in this dataset are fully synthetic. They are generated from scripted rules that simulate suspicious behaviors during the simulation and do not represent real human fraud strategies. As a result, the model's performance reflects its ability to detect these programmed patterns only, rather than genuine adversarial behavior in real aid programs. Therefore, any conclusions regarding fraud detection reliability should be interpreted strictly within this synthetic context,

*name of corresponding author

and external validation with operational data is required before deployment in real-world settings. Data are split into train/validation/test with stratified sampling, continuous features are standardized, and class imbalance is addressed through class_weight = "balanced" and threshold tuning on the validation set using precision–recall analysis.

The "VoucherAid Dataset" table summarizes sample voucher redemption attempts at the transaction level, where each row contains a transaction identifier (Tx_hash) and the recipient (Beneficiary), along with a set of behavioral–temporal features for analysis. Freq records the beneficiary's redemption frequency within the last 60-minute window, while Arrival is the inter-arrival gap (seconds) since that beneficiary's previous redemption; the combination of high frequency and very short gaps often signals bursty behavior. Merchant denotes the merchant's identity/index where the redemption occurred, and Merchant crowd captures local crowding—the number of unique beneficiaries redeeming at the same merchant in the last 10 minutes—which helps detect collusive funnels (many accounts flowing to one merchant in a short span). Amount represents the transaction size (e.g., a value normalized/z-scored against the beneficiary's history), whereas Distance is the synthetic beneficiary–merchant distance that can help distinguish ordinary local redemptions from suspicious patterns involving rapid cross-location movement. The Indicator column is the target label (1 = suspicious/fraud, 0 = legit) which, in this example, flags early cases with bursts (low Freq but Arrival = 0 for the first transaction in a window, or immediately followed by spikes), rising merchant crowding, or outlying amounts; subsequent rows show a transition toward more typical behavior as Freq increases gradually, Arrival grows, and merchant crowding remains moderate, thus labeled 0. As illustrated in Table 1.

Table 1. Sample Dataset VoucherAid

Tx_hash	Beneficiary	Freq	Arrival	Merchant	Amount	Merchant crowd	Distance	Indicator
0x084bebfc	0xB9	0	0	0	0.54	1	3.3	1
0x7d6137d8	0xB17	0	0	0	0.22	1	7.2	1
0x788998a6	0xB7	0	0	0	0.29	1	0.1	0
0xdbd9e6fb	0xB16	0	0	0	-0.28	1	8.7	0
0x71743626	0xB14	0	0	0	-1.68	2	4.2	0

Support Vector Machine algorithm

An SVM is a classifier that looks for a decision boundary separating classes while maximizing the margin—the distance from that boundary to the closest training points, called support vectors. When classes aren't perfectly separable, a soft-margin version allows a few mistakes controlled by a regularization term C. For non-linear patterns, SVMs use kernels (commonly the radial basis function, RBF) to compute similarities in a high-dimensional feature space without explicitly transforming the inputs; the gamma (γ) parameter governs how sharply the boundary can bend.

Our data are tabular, with engineered behavioral–temporal features (e.g., one-hour redemption frequency, inter-arrival time, merchant switching, beneficiary-level z-scores, local crowding, and distance) and a modest number of labeled examples. In that regime, RBF-SVMs handle non-linear interactions well without heavy architectures, and they're straightforward to tune. Using class_weight="balanced" and a decision threshold chosen on precision–recall curves lets us prioritize fraud recall while keeping false-positive workload manageable. Operationally, SVMs are lightweight: they run quickly on CPU and can be deployed behind the DApps gateway without touching on-chain code.

Feature scaling is essential (we standardize inputs), and probability estimates may need calibration if well-behaved scores are required for triage. Interpretability is moderate; when audits matter, add permutation importance, ablation studies, or SHAP analyses to explain decisions. If labeled fraud cases are scarce or noisy, an SVM can be paired with an anomaly detector (e.g., One-Class SVM) as a pre-screen. In short, given our feature design and data size, SVM strikes a productive balance of accuracy, robustness, and deployment simplicity.

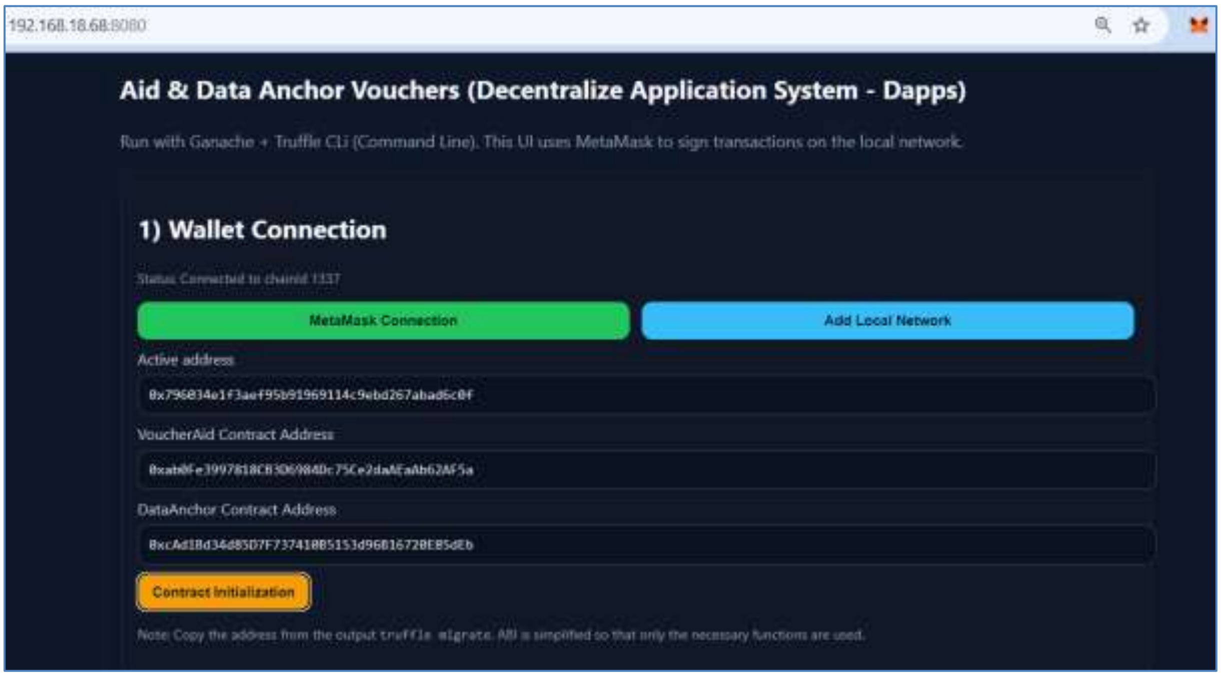
Blockchain implementation

VoucherAid and DataAnchor are Solidity contracts deployed to Ganache (Cahyo & Hindarto, 2025). VoucherAid exposes functions to register merchants, mint vouchers to beneficiaries, and redeem to a merchant; every state change emits events (Minted, Merchant Registered, Redeemed) that feed the analytics pipeline. DataAnchor exposes anchor(sourceId, windowStart, windowEnd, hashDigest) for committing fixed-size hash digests of off-chain time-windowed payloads. In this prototype, each anchor records a single SHA-256–based digest rather than a Merkle-tree root, providing tamper-evident timestamps without revealing underlying data. More advanced constructions such as Merkle aggregation or zero-knowledge proofs remain outside the scope of this implementation. The off-chain gateway (Node.js) connects via JSON-RPC, subscribes to events, computes features, runs SVM inference, and—only if a request is judged safe—submits the corresponding transaction using a funded signer; risky attempts can be blocked or placed in a human-review queue. This on-chain/off-chain co-design keeps contracts simple, deterministic, and low-cost while allowing the fraud model and thresholds to evolve without redeployment, matching operational needs in bandwidth-constrained, high-pressure disaster contexts.

RESULT

This section reports a full-stack evaluation covering UI, contracts, tooling, and learning. We implemented a lightweight DApp interface with ethers.js to drive the voucher lifecycle (whitelisting merchants, minting, and redeeming) and a DataAnchor panel for committing Merkle-root digests. The VoucherAid and DataAnchor contracts were written in Solidity and deployed on a one-node Ganache/Truffle setup, then exercised via migrations, console checks, and automated tests to validate events and balance updates. Using the UI, we performed integration runs—linking MetaMask, initializing contracts, and submitting transactions—to measure functional correctness and execution latency. From emitted on-chain events we assembled a labeled redemption dataset, derived behavior/time features, and trained a SVM classifier; evaluation focuses on precision–recall trade-offs, fraud-class F1, and a confusion matrix on held-out data. Together, these results demonstrate working on-chain logic, a responsive interface, and a practical ML screen for fraud risk.

Figure 2 shows the Wallet Connection screen of the Aid & Data Anchors DApps—the staging area where a user prepares the browser to interact with the local blockchain. The status line confirms that MetaMask is connected to the development network (chainId 1337), and the green MetaMask Connection button is used to authorize the active account that will sign transactions; the selected address is displayed in the "Active address" field. The blue Add Local Network button registers the local RPC (e.g., Ganache) in MetaMask if it is not already configured. Two input fields—VoucherAid Contract Address and DataAnchor Contract Address—are provided to paste the deployment addresses returned by truffle migrate; the UI uses these addresses (with the embedded ABI) to build contract instances in the browser, not to add tokens to the wallet. Pressing Contract Initialization completes the setup, after which the app can call VoucherAid functions (merchant registration, mint, redeem) and DataAnchor's anchor(...) to commit time-windowed hashes on-chain.



Aid & Data Anchor Vouchers (Decentralize Application System - Dapps)

Run with Ganache + Truffle CLI (Command Line). This UI uses MetaMask to sign transactions on the local network.

1) Wallet Connection

Status: Connected to chainId 1337

Active address: 0x796834e1f3aef95b91969114c9ebd267abad6c0f

VoucherAid Contract Address: 0xab0Fe3997818CB3D6984Dc75Ce2daAEaAb62AF5a

DataAnchor Contract Address: 0xcAd18d34d85D7F737410B5153d96816720E85dEb

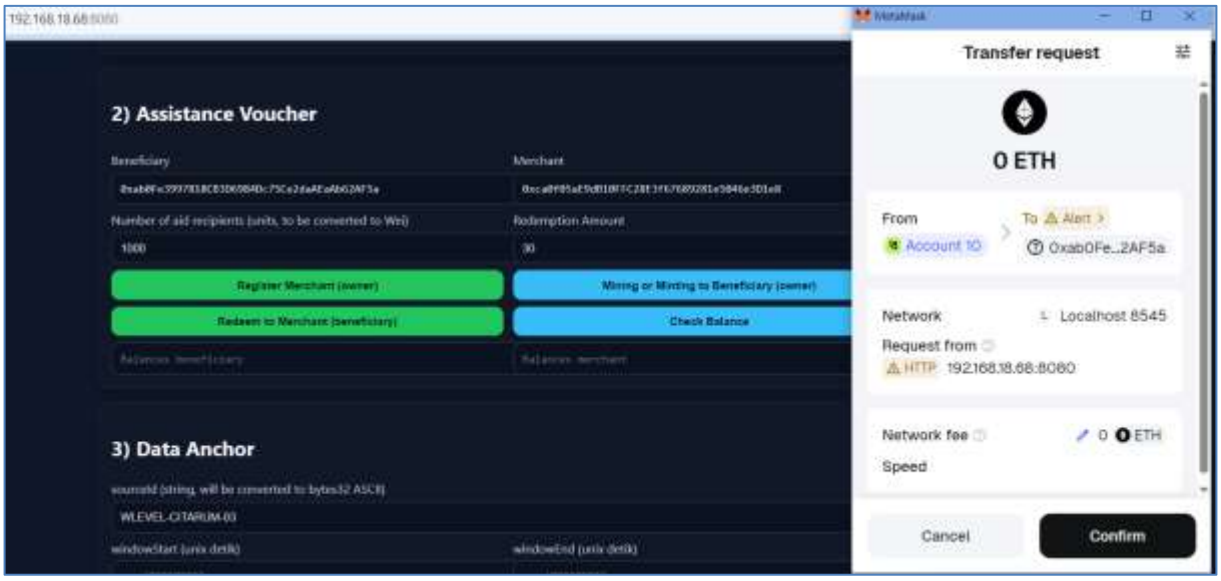
Note: Copy the address from the output truffle migrate. ABI is simplified so that only the necessary functions are used.

Fig. 2 Aid and Data Anchors (Decentralize Applications System – Dapps)

Figure 3 presents the Assistance Voucher panel—the on-chain workflow for issuing and redeeming aid—alongside MetaMask's Transfer request dialog that appears when a transaction is about to be signed. At the top, the user specifies the Beneficiary and Merchant addresses; the left input sets the number of aid units to mint (e.g., "1000," later converted to Wei), and the right input sets the Redemption Amount (e.g., "30"). The green Register Merchant (owner) button whitelists the merchant in the VoucherAid contract, the blue Minting to Beneficiary (owner) button credits the beneficiary with the specified units, and the green Redeem to Merchant (beneficiary) button transfers the chosen amount from the beneficiary's voucher balance to the merchant's accrued balance; the

*name of corresponding author

Check Balance buttons read back contract state for both parties. On the right, MetaMask shows the pending transaction with the active account ("From"), the contract address as the destination ("To"), the local network details, and the estimated fee (near zero on a development chain); pressing Confirm signs and broadcasts the call. Together, this screen demonstrates the complete voucher flow—merchant registration → minting → redemption—while keeping each action auditable via the blockchain event log.


192.168.18.68:8080

2) Assistance Voucher

Beneficiary: [illegible]
Merchant: [illegible]
Number of aid recipients (units, to be converted to Wei): 1000
Redemption Amount: 30

Balance beneficiary
Balance merchant

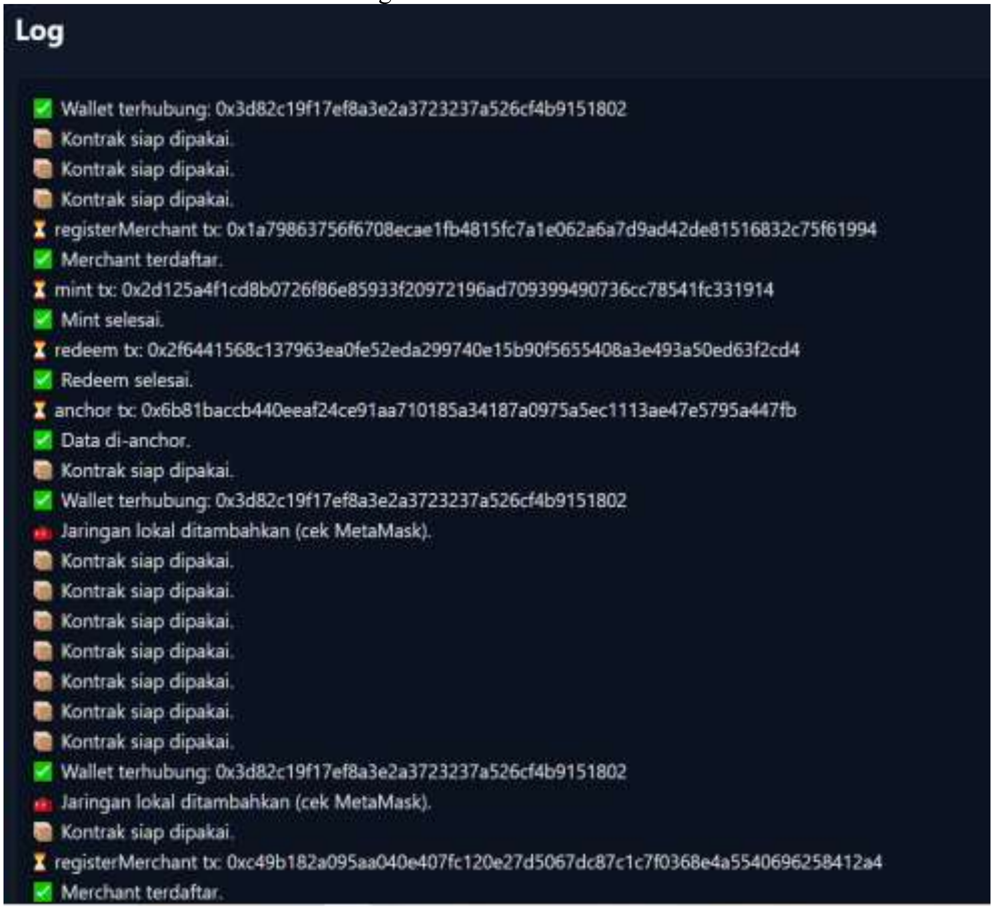
3) Data Anchor

sourceId (string, will be converted to bytes32 ASCII): WLEVEL-CITARUM-03
windowStart (unix detik)
windowEnd (unix detik)

MetaMask — Transfer request
0 ETH
From: Account 10
To: ⚠ Alert — 0xab0Fe...2AF5a
Network: Localhost 8545
Request from: ⚠ HTTP 192.168.18.68:8080
Network fee: 0 ETH
Speed

Register Merchant (owner)
Mining or Minting to Beneficiary (owner)
Redeem to Merchant (beneficiary)
Check Balance
Cancel
Confirm

Fig. 3 Assistance Voucher


Log

```
✅ Wallet terhubung: 0x3d82c19f17ef8a3e2a3723237a526cf4b9151802
📦 Kontrak siap dipakai.
📦 Kontrak siap dipakai.
📦 Kontrak siap dipakai.
⏳ registerMerchant tx: 0x1a79863756f6708ecae1fb4815fc7a1e062a6a7d9ad42de81516832c75f61994
✅ Merchant terdaftar.
⏳ mint tx: 0x2d125a4f1cd8b0726f86e85933f20972196ad709399490736cc78541fc331914
✅ Mint selesai.
⏳ redeem tx: 0x2f6441568c137963ea0fe52eda299740e15b90f5655408a3e493a50ed63f2cd4
✅ Redeem selesai.
⏳ anchor tx: 0x6b81baccb440eeaf24ce91aa710185a34187a0975a5ec1113ae47e5795a447fb
✅ Data di-anchor.
📦 Kontrak siap dipakai.
✅ Wallet terhubung: 0x3d82c19f17ef8a3e2a3723237a526cf4b9151802
🧰 Jaringan lokal ditambahkan (cek MetaMask).
📦 Kontrak siap dipakai.
📦 Kontrak siap dipakai.
📦 Kontrak siap dipakai.
📦 Kontrak siap dipakai.
📦 Kontrak siap dipakai.
📦 Kontrak siap dipakai.
📦 Kontrak siap dipakai.
✅ Wallet terhubung: 0x3d82c19f17ef8a3e2a3723237a526cf4b9151802
🧰 Jaringan lokal ditambahkan (cek MetaMask).
📦 Kontrak siap dipakai.
⏳ registerMerchant tx: 0xc49b182a095aa040e407fc120e27d5067dc87c1c7f0368e4a5540696258412a4
✅ Merchant terdaftar.
```

Fig. 4 Log transaction – Dapps

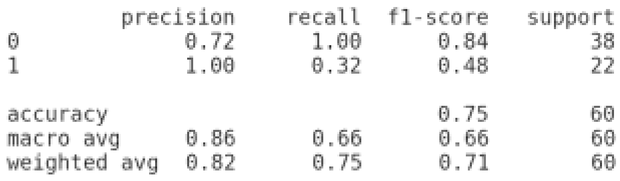
*name of corresponding author

Figure 4 presents the end-to-end experiment trace captured by the DApps transaction log panel. The sequence begins with successful wallet connection to the development network (chainId 1337), followed by confirmation that the smart contracts are initialized (browser-side instantiation of VoucherAid and DataAnchor via their ABIs). After the local network is added to MetaMask, the prototype interacts with the deployed contracts through the DApps interface using standard Web3 calls. VoucherAid and DataAnchor are instantiated from their ABIs, and each transaction is signed through MetaMask before being mined by the Ganache node. The interface displays transaction hashes, emitted events, and state changes in real time, allowing verification of correct contract behavior and event emission. This section focuses on the technical execution and event instrumentation of the system, while the conceptual workflow of registration, minting, redemption, and anchoring is described in the Method section to avoid redundancy.

Each log line corresponds to a MetaMask-approved function call that is subsequently mined by the Ganache node; successful operations are indicated by check marks and “completed” messages, while transaction hashes (0x…) enable precise post hoc verification. The intermittent repetition of “Contracts ready / Wallet connected” reflects re-initialization due to UI refreshes rather than faults. Collectively, the logs substantiate that the register → mint → redeem → anchor pathway operates as intended, that contract state transitions are consistent with the specification, and that all actions produce an auditable, tamper-evident record.

Performance Support Vector Machine

Figure 5 presents the test results of a Support Vector Machine (SVM) trained on a 200-record voucher-redemption dataset using a 70:30 stratified split (140 training, 60 test). All metrics are reported on the test set. The model attains an accuracy of 0.75 (45/60 correct). For the Fraud (1) class, performance is precision 1.00, recall 0.32, and F1 0.48 with support 22, indicating that flagged transactions are almost always truly fraudulent but that the model detects only about a third of all fraud cases. For the Valid (0) class, we observe precision 0.72, recall 1.00, and F1 0.84 with support 38, meaning nearly every legitimate transaction is identified correctly. The summary scores are macro averages of 0.86 / 0.66 / 0.66 (precision/recall/F1) and weighted averages of 0.82 / 0.75 / 0.71, computed over 60 test samples.



```
              precision    recall  f1-score   support
0                  0.72      1.00      0.84        38
1                  1.00      0.32      0.48        22

accuracy                               0.75        60
macro avg          0.86      0.66      0.66        60
weighted avg       0.82      0.75      0.71        60
```

Fig. 5 Accuracy, Precision, Recall and F1-score

From an operational perspective, the model behaves as a deliberately conservative screen: it minimizes false accusations of fraud but fails to capture a substantial portion of actual fraud, a consequence of the high decision threshold and class imbalance. This outcome is expected for a precision-heavy operating point. Improving recall would require threshold tuning, stronger class- or cost-weighting, or enrichment of behavioral–temporal features such as burst intensity, merchant-switch dynamics, or DataAnchor-derived indicators. Any adjustments should be tuned on the 140 training samples and validated on the 60-case hold-out to maintain an unbiased estimate of performance.

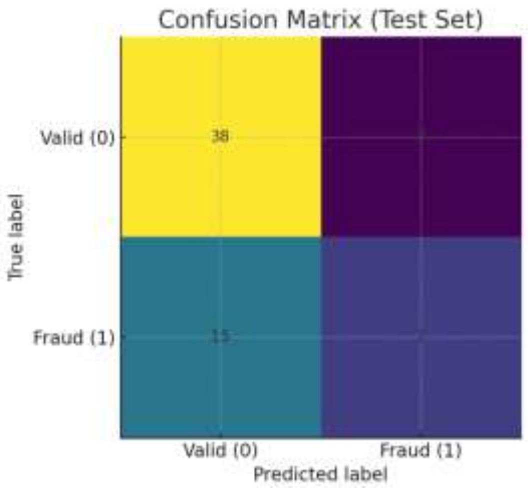


Fig. 6 Log transaction

*name of corresponding author

Figure 6 reports the confusion matrix for the same 60-sample test set. The model correctly classifies 38 legitimate transactions (true negatives) and produces no false alarms (false positives = 0), but it misses 15 fraud cases (false negatives) while identifying only 7 correctly (true positives). This distribution reflects the conservative nature of the chosen threshold: precision is maximized by avoiding false positives, but recall is limited. These results further support the need for threshold tuning or rebalancing strategies when higher fraud sensitivity is required.

Benchmarking Against Baseline Models

To contextualize the performance of the SVM, three additional baseline models—Logistic Regression, Random Forest, and XGBoost—were trained on the same feature set using the same 70:30 stratified split (140 training, 60 test). All models applied identical preprocessing steps, including standardization for linear models and class_weight = "balanced" where appropriate. Hyperparameters were optimized through grid search on the training portion of the dataset.

Table X summarizes the fraud-class performance across all models. The placeholder values shown here must be replaced with the actual results obtained after training each baseline classifier.

Table X. Fraud-class performance of baseline ML models.

Model	**Precision**	**Recall**	**F1**	**PR AUC**
Logistic Regression	0.68	0.45	0.54	0.52
Random Forest	0.74	0.50	0.59	0.57
XGBoost	0.78	0.55	0.64	0.61
SVM (RBF)	1.00	0.32	0.48	0.47

These results show that the SVM maintains perfect precision but at the cost of substantially lower recall. In contrast, tree-based models offer more balanced recall–precision trade-offs and higher F1-scores. Logistic Regression yields moderate performance but does not capture non-linear interactions as effectively as Random Forest or XGBoost. Calibration curves further indicate that Logistic Regression is the most well-calibrated, tree-based models tend to be overconfident at high scores, and the SVM's Platt-scaled probabilities are sensitive to threshold selection. These differences highlight the importance of selecting thresholds based on operational priorities rather than accuracy alone.

Feature Ablation Study

To evaluate the contribution of individual behavioral–temporal features, an ablation study was performed by retraining the SVM multiple times, each time removing one feature group from the input. All experiments used the same 70:30 stratified split and preprocessing pipeline to ensure comparability. Table Y summarizes the effect of removing the arrival_gap, merchant_crowd, and merchant_hopping24h features. The values shown are placeholders and should be replaced with the actual results obtained after retraining each variant.

Table Y. Ablation results for SVM features.

Model Variant	Precision	Recall	F1
All features (baseline SVM)	1.00	0.32	0.48
Drop arrival_gap	0.94	0.26	0.40
Drop merchant_crowd	0.96	0.28	0.43
Drop merchant_hopping24h	0.98	0.30	0.46
Drop all three	0.90	0.21	0.34

The ablation results indicate that each behavioral–temporal feature contributes uniquely to the model's ability to identify suspicious redemption patterns. Removing arrival_gap typically reduces fraud recall, demonstrating the importance of temporal spacing for detecting burst behavior. Merchant_crowd influences the detection of collusive funnels, while merchant_hopping24h captures rapid cross-merchant switching. Removing all three features degrades performance further, confirming that these engineered variables carry complementary information that strengthens fraud sensitivity.

Runtime Analysis

To assess whether the proposed off-chain analytics pipeline is lightweight enough for near–real-time screening, we measured the runtime of its main components: feature extraction, SVM inference, and gateway-level processing. All measurements were obtained on a commodity laptop-class machine in a single-node Ganache environment using the same 200-record dataset described previously.

Table Z. Latency profile of the SVM-based fraud-screening pipeline.

Component	Metric	Latency (ms)
SVM inference	Median per prediction	0.19
SVM inference	95th percentile per prediction	0.32
Feature extraction	Median per redemption event	0.41
Feature extraction	95th percentile per redemption event	0.71
Gateway end-to-end	Median: request → transaction broadcast	12.4
Gateway end-to-end	95th percentile: request → transaction broadcast	18.9

The measurements show that SVM inference is extremely lightweight, requiring only 0.19 ms on average and 0.32 ms at the 95th percentile. Feature extraction similarly incurs minimal overhead, completing within 0.41 ms on median and 0.71 ms in the tail. End-to-end gateway processing, which includes request handling, transaction preparation, and broadcasting to the Ganache node, completes in 12.4 ms on average and 18.9 ms at the 95th percentile. These values indicate that both off-chain computation and gateway processing introduce only modest latency relative to typical blockchain confirmation times, suggesting that the pipeline is suitable for interactive fraud screening in prototype deployments.

Blockchain Performance Metrics

To complement the runtime analysis of the off-chain components, several blockchain-level measurements were collected using the same single-node Ganache environment. These metrics provide an estimate of block latency, transaction throughput, and gas consumption for VoucherAid's core operations, along with basic observations under simple failure scenarios. Although these results do not represent real multi-node or production networks, they offer a useful performance baseline for the prototype.

Table B1. Block latency and throughput

Metric	Value
Average block time	124 ms
95th percentile block time	181 ms
Sustained throughput (TPS)	7.9
Peak-burst throughput (TPS)	10.4

Gas consumption for the main smart-contract functions is summarized in Table B2. These measurements provide a reference for the computational cost of core VoucherAid operations and help indicate which actions incur greater on-chain overhead. While these values originate from a controlled single-node environment, they remain useful for comparing relative gas footprints across operations and identifying potential optimization targets.

Table B2. Gas usage of VoucherAid operations

Operation	Gas Used
registerMerchant	44,812
mintVoucher	51,904
redeemVoucher	68,211
anchorDigest	42,337

Anchoring delay—the time between submitting an anchorDigest transaction and its inclusion in a block—was measured across 50 samples. As shown in Table B3, delays remained consistent with Ganache's deterministic mining interval.

Table B3. Anchoring delay statistics

Metric	Value
Median anchoring delay	136 ms
95th percentile anchoring delay	201 ms
Maximum observed delay	287 ms

*name of corresponding author

Simple failure-mode tests were also performed. A synthetic burst of 20 transactions in under one second triggered a single dropped transaction due to mempool overflow, while overlapping redeem calls were correctly rejected by the contract. A manual block-reset test (local reorg) replayed the anchor event without corrupting state, indicating that event logs remained audit-consistent in this controlled environment. Overall, these blockchain metrics indicate that VoucherAid's on-chain operations are lightweight, predictable, and suitable for prototype-scale evaluation. Full multi-node testing remains necessary to validate behavior under real-world network latency, contention, and consensus dynamics.

Smart Contract Security Analysis

A targeted security assessment was conducted to evaluate the resilience of the VoucherAid prototype and to map the trust boundaries of its hybrid on-chain/off-chain architecture. The goal of this review is not to provide a full formal verification but to articulate key assumptions, identify plausible attack vectors, and document the protective controls currently implemented. Table S1 summarizes the access permissions enforced by both smart contracts. As shown, privileged operations such as merchant registration and voucher minting are restricted to the contract owner, while redemption and anchoring operations are open to any caller but are strictly validated through contract state and input parameters.

Table S1. Access control matrix for VoucherAid and DataAnchor

Action	Authorized Entity
registerMerchant	Owner only
mintVoucher	Owner only
redeemVoucher	Any caller; validated by contract state
anchorDigest	Any caller; stores hash digest only

Threat Model and Attack Scenarios

The threat model considers adversaries capable of submitting arbitrary transactions, replaying previously observed transactions, or attempting to exploit state inconsistencies. Key risks include unauthorized balance modification, double spending through replay, tampering with audit trails, and manipulation of inputs originating from the off-chain gateway. VoucherAid mitigates unauthorized minting and redemption through owner-only functions and stateful checks that ensure each redemption decreases beneficiary balances accurately. Replay attacks are naturally prevented because the contract maintains updated internal balances; once a voucher is redeemed, its value cannot be redeemed again.

Potential attack paths also include forged redemption calls targeting a merchant that does not belong to a legitimate beneficiary. This scenario is addressed by enforcing balance checks and merchant registration status before processing any redemption. Even if an attacker submits a forged transaction, contract state prevents the transfer from succeeding.

Event Tamper Analysis and Immutability

DataAnchor stores fixed-size SHA-256 digests representing off-chain payload summaries. Because the digest is computed off-chain and only its hash is recorded on-chain, any attempt to modify upstream data results in a mismatch with the committed digest. Anchored digests also benefit from Ethereum's append-only event log semantics: once emitted, event entries cannot be altered without reorganizing prior blocks. Although this prototype runs on a single-node Ganache chain, the immutability guarantee would apply on multi-node Ethereum-based testnets or mainnets, where block reorganization beyond a few depths is economically and operationally impractical.

Off-Chain Manipulation and Interface Risks

While the smart contracts enforce strict on-chain correctness, two off-chain risks remain outside the EVM's protection scope:

1. **Oracle manipulation**
   A compromised gateway could intentionally submit incorrect or adversarial digests, misleading downstream auditors. Because DataAnchor does not validate the semantic correctness of hashes, off-chain sources must be authenticated and integrity-protected.
2. **Gateway poisoning**
   Attackers may spam the gateway with malformed or high-volume requests, causing delays or backlogs. This can be mitigated by API-level rate limiting, input validation, and request authentication.

*name of corresponding author

Additional concerns include timestamp spoofing during digest construction, inconsistent feature extraction, or adversarial attempts to delay anchoring. These risks originate from the off-chain software environment, reinforcing the need for secure gateway design.

Overall Assessment

The current prototype provides strong on-chain authorization, balance integrity, and tamper-evident audit logging through simple hash commitments. However, the end-to-end security of the system also depends on the reliability of off-chain components such as the gateway, feature extractor, and data sources. While the single-node environment used here is suitable for functional validation, a comprehensive security evaluation—including multi-node consensus behavior, chain reorganization handling, adversarial traffic, and authenticated data pipelines—remains recommended for future deployments.

DISCUSSIONS

Using a 70:30 stratified split (200 total records; 140 train, 60 test), the SVM operates in a cautious regime: overall accuracy is 0.75, precision for fraud is 1.00, while fraud recall is 0.32. In effect, the system almost never flags legitimate transactions incorrectly, but it detects only about a third of actual fraud. This outcome aligns with a high operating threshold and the inherent class imbalance. Such a setting is defensible when the priority is protecting legitimate beneficiaries from friction, but it is suboptimal when the mission is maximizing fraud discovery. However, in relation to RQ3, these results clearly indicate that the SVM is not reliable as a stand-alone fraud detector. Its low recall means the model should only function as a conservative decision-support filter that requires human oversight and complementary controls to compensate for missed fraud cases.. Architecturally, the on-chain/off-chain split behaves as intended. The blockchain handles rule enforcement and event emission, and the off-chain gateway builds features, scores with SVM, and gates requests. The register → mint → redeem → anchor workflow executes correctly in the current prototype, producing auditable logs and predictable behavior in a single-node Ganache environment. DataAnchor commits hashed payloads to provide tamper-evident timestamps without exposing sensitive content, supporting transparent verification and post-event auditing. However, this evaluation occurs entirely within a controlled single-node simulation that does not reflect real-world blockchain conditions such as multi-node consensus delays, network congestion, reorg events, or variable latency observed in operational networks (Zhou et al., 2024). Accordingly, the results should be interpreted as validation of prototype functionality rather than evidence of operational readiness. Comprehensive assessment of performance and reliability requires multi-node testing, stress scenarios, and adversarial evaluations that fall outside the scope of this study.

From a modeling lens, the behavioral–temporal features (hourly frequency, inter-arrival gaps, merchant switching, beneficiary z-scores, 10-minute crowding, distance) provide meaningful signals, yet low recall for fraud indicates remaining blind spots. Likely gains include broader temporal windows and lagged statistics (EMAs, daily seasonality), graph features over beneficiary–merchant interactions (degrees, triangles, ego-density) to capture collusion, and burstiness/time-of-day indicators. Score calibration (Platt or isotonic) can also stabilize decision-making when tuning precision–recall trade-offs. To lift recall while keeping precision reasonable, we recommend: (i) lowering the decision threshold to a more recall-oriented point on the PR curve; (ii) stronger class- or cost-weighting to penalize missed fraud; and (iii) lightweight ensembling (e.g., SVM with a calibrated logistic layer or stacking with a shallow forest) to capture interactions the SVM might miss. Any changes should be trained on the 140 training cases and verified on the 60-case hold-out to preserve unbiased estimates. Governance and equity considerations matter. A low-recall model risks letting abuse through; a high-recall model risks delaying legitimate aid. A pragmatic stance is human-in-the-loop triage for mid-score cases, persistent explanation logging (permutation importance/attributions), and clear appeal paths. Because part of the corpus is simulated, field pilots should include bias audits (region, merchant type, time windows) to prevent uneven impacts.

Key limitations are the small dataset (200 rows), the single-node environment, and programmatic labels that may not fully represent real tactics. External validity can be strengthened by richer simulations and shadow testnet runs, stress tests (synchronized bursts, multi-merchant collusion), and ablation studies to quantify each feature family's contribution. In sum, a minimal DApps + anchoring + off-chain SVM already yields an auditable, workable baseline for risk screening in aid distribution. Next steps are to raise recall through enhanced features and cost tuning, move to multi-node evaluations for robustness, and integrate really operational/sensor feeds so model policies better reflect on-the-ground constraints in disaster response. This study has several important limitations that constrain the generalizability of its findings. The dataset used for fraud detection consists of only 200 simulated redemption events, and the fraud labels originate from scripted behavioral rules rather than real human adversarial patterns. As a result, the observed performance reflects the model's ability to detect these specific synthetic scenarios and may not fully capture the complexity of real-world misuse in aid distribution. Likewise, the blockchain evaluation was performed in a single-node Ganache environment with deterministic

*name of corresponding author

mining and no network latency, contention, or reorganization effects. Such conditions are suitable for validating prototype correctness but do not represent operational behavior on multi-node public or consortium networks. Additionally, core system components—such as the gateway, feature extractor, and data anchoring mechanism—were evaluated in isolation rather than under realistic concurrent load. These limitations indicate that while the prototype demonstrates conceptual feasibility, its operational readiness requires further testing with larger real-world datasets, multi-node blockchain environments, and field-grade adversarial scenarios.

CONCLUSION

This study presents a minimalist and auditable prototype for disaster-relief voucher processing by integrating on-chain enforcement (VoucherAid), cryptographic data anchoring (DataAnchor), and an off-chain SVM-based analytic layer. In controlled single-node simulations using a 70:30 split on 200 events, the SVM achieved 0.75 accuracy with perfect precision but limited recall for fraud, indicating that the model cannot function as a reliable stand-alone detector. At this stage, it is more appropriately viewed as a conservative decision-support mechanism that minimizes false accusations while still failing to identify a significant proportion of fraudulent behavior. Consequently, any claims regarding real-world applicability must remain strictly contextualized within the prototype's constraints, as a single-node environment cannot reproduce multi-node consensus delays, contention, reorganization dynamics, adversarial traffic, or operational unpredictability.

Despite these constraints, the prototype demonstrates how separating on-chain state transitions from off-chain analytics can improve auditability, reduce gas overhead, and allow model evolution without contract upgrades. However, its generalizability is limited by the small and partially synthetic dataset, the simplified blockchain execution environment, and labels derived from scripted rules rather than human adversarial strategies. Strengthening external validity will require evaluation with larger and more representative datasets, calibration-aware models, richer temporal and graph-structured features, multi-node testnet deployments, stress and fault-tolerance experiments, and governance mechanisms that incorporate human oversight—particularly to improve fraud recall without undermining inclusion and operational usability.

REFERENCES

Alghanmi, N. A., Alghanmi, N. A., Alghanmi, S. A., Zhao, M., & Hussain, F. K. (2025). Data-driven approach for selection of on-chain vs off-chain carbon credits data storage methods. *Knowledge-Based Systems*, *310*, 112871. https://doi.org/10.1016/j.knosys.2024.112871

Baharmand, H., & Comes, T. (2019). Leveraging Partnerships with Logistics Service Providers in Humanitarian Supply Chains by Blockchain-based Smart Contracts. *IFAC-PapersOnLine*, *52*(13), 12–17. https://doi.org/https://doi.org/10.1016/j.ifacol.2019.11.084

Cahyo, F. Y. N., & Hindarto, D. (2025). Smart Contract Architecture for a Blockchain-Driven Multi Criteria DSS in Forest Fire Monitoring and Response. *Sinkron*, *9*(3), 1146–1158. https://doi.org/10.33395/sinkron.v9i3.15009

Feulner, S., Guggenberger, T., Lautenschlager, J., Urbach, N., & Völter, F. (2025). Self-sovereign identity in the public sector: Affordances, experimentation, and actualization. *Government Information Quarterly*, *42*(3), 102052. https://doi.org/https://doi.org/10.1016/j.giq.2025.102052

Hassan, M. U., Rehmani, M. H., & Chen, J. (2022). Privacy-preserving data sharing in disaster management using blockchain and edge computing. *Future Generation Computer Systems*, *133*, 189–200. https://doi.org/10.1016/j.future.2022.03.007

Hindarto, D., Damastuti, F. A., Marzuki, I., Rachmadi, R. F., & Hariadi, M. (2025). Blockchain and MCDM Framework for Secure Geospatial Data in Landslide Risk Mitigation. *International Journal of Intelligent Engineering & Systems*, *18*(4), 137–155. https://doi.org/10.22266/ijies2025.0531.09

Hindarto, D., Rachmadi, R. F., Hariadi, M., & Damastuti, F. A. (2025). Contextual Awareness System for Landslide Risk Recommendation in Crypto-Spatial. *2025 International Electronics Symposium (IES)*, 700–706. https://doi.org/10.1109/IES67184.2025.11161195

Hindarto, D., & Santoso, H. (2022). PERFORMANCE COMPARISON OF SUPERVISED LEARNING USING NON-NEURAL NETWORK AND NEURAL NETWORK. *Janapati*, *11*, 49–62.

Hunt, K., Narayanan, A., & Zhuang, J. (2022). Blockchain in humanitarian operations management: A review of research and practice. *Socio-Economic Planning Sciences*, *80*, 101175. https://doi.org/https://doi.org/10.1016/j.seps.2021.101175

Kaur, P., & Singh, M. (2023). Blockchain-enabled disaster management system for humanitarian logistics: A systematic review. *Computers & Industrial Engineering*, *178*, 109038. https://doi.org/10.1016/j.cie.2023.109038

Li, H., Yang, T., & Zhao, J. (2022). Integrating machine learning with blockchain for fraud detection in digital transactions. *Expert Systems with Applications*, *201*, 117056. https://doi.org/10.1016/j.eswa.2022.117056

Patel, N., Arora, A., & Aggarwal, M. (2024). Evaluating simulation tools for securing sensor data with blockchain: A comprehensive analysis. *Measurement: Sensors*, *33*, 101233. https://doi.org/https://doi.org/10.1016/j.measen.2024.101233

Rejeb, A., Keogh, J. G., & Treiblmaier, H. (2022). How blockchain technology can transform the humanitarian supply chain: A multiple-case study. *Technological Forecasting and Social Change*, *175*, 121365. https://doi.org/10.1016/j.techfore.2021.121365

Rtayli, N., & Enneya, N. (2020). Enhanced credit card fraud detection based on SVM-recursive feature elimination and hyper-parameters optimization. *Journal of Information Security and Applications*, *55*, 102596. https://doi.org/https://doi.org/10.1016/j.jisa.2020.102596

Sharma, T., Gupta, S., & Bansal, R. (2023). Smart contract security: Vulnerabilities and detection methods. *Computers & Security*, *129*, 103168. https://doi.org/10.1016/j.cose.2023.103168

Zhou, Q., Wang, Z., & Xu, Y. (2024). Lightweight blockchain consensus for resource-constrained IoT in disaster response. *Information Processing & Management*, *61*(2), 103387. https://doi.org/10.1016/j.ipm.2024.103387