Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Compliance-Aware Payment Gateway Operations for Healthcare and Medical Device Ecosystems: A DevOps Framework for Resilient Digital Transaction Systems

Vimal Teja Manne[1] and Sudhakavya Bodapati Venkata[2]
[1] *The University of Texas at Dallas, Richardson, United States.*
[2] *The University of Texas at Dallas, Richardson, United States.*
Corresponding Author: Vimal Teja Manne, E-mail: vimalteja.m@gmail.com

| ABSTRACT
Healthcare and medical device ecosystems increasingly depend on reliable digital transaction sys-tems for patient payments, provider collections, device-service billing, repair fees, warranty adjustments, refunds, reversals, chargeback evidence, settlement, and reconciliation. In such environments, payment gateway failures are not only technical incidents but also business, operational, and compliance risks. This paper proposes a compliance-aware DevOps framework for resilient digital transaction systems in healthcare and medical device ecosystems. The framework integrates payment gateway operations, transaction lifecycle monitoring, DevOps observability, incident response automation, business-impact scoring, and compliance evidence mapping. A simulation-based evaluation is designed using the PaySim synthetic financial transaction dataset as the base transaction layer. Synthetic healthcare workflow labels, payment gateway operational attributes, DevOps telemetry, and compliance evidence fields are gener-ated through controlled and reproducible rules. Six failure scenarios are injected: authorization timeout, refund failure, reversal delay, settlement mismatch, gateway latency spike, and deployment-induced transaction failure. The proposed framework is compared with two baselines: generic infrastructure monitoring and payment-only monitoring. Across repeated simulation runs, the proposed framework reduces mean time to detect and mean time to recover while improving reversal completion, refund completion, reconciliation handling, and compliance evidence completeness. The study contributes a reproducible business-DevOps model for mapping payment lifecycle exceptions to observability signals, recovery runbooks, business-impact scores, and compliance evidence controls in regulated transaction environments.

| KEYWORDS
Payment gateway; healthcare payments; medical device systems; DevOps; observability; transaction resilience; PaySim; compliance; refund; reconciliation

| ARTICLE INFORMATION
ACCEPTED: 04 February 2024 **PUBLISHED:** 25 February 2024 **DOI:** 10.32996/jcsts.2024.6.1.37

1. INTRODUCTION

Digital transaction systems have become a critical operational component of healthcare and medi-cal device ecosystems. Healthcare providers, service networks, repair centers, device manufacturers, warranty-support teams, patient-facing portals, and provider payment operations increasingly depend on payment gateways to support authorization, capture, refund, reversal, chargeback evidence, settle-ment, and reconciliation workflows. Unlike ordinary transaction systems, healthcare and medical device transaction workflows may be connected to patient billing corrections, device repair orders, replacement devices, warranty adjustments, service subscriptions, and provider-facing reconciliation.

A payment gateway in a regulated ecosystem must support more than basic transaction approval.

It must preserve traceability across the transaction lifecycle, minimize payment-data exposure, support timely refunds and reversals, maintain settlement consistency, and provide evidence for operational and compliance review. If an authorization request times out, the organization may not immediately know whether the transaction was approved, declined, or left in an unknown state. If a reversal is delayed, duplicate payment risk and customer support pressure may increase. If refund or chargeback evidence is incomplete, dispute handling can become slower and more privacy-sensitive. If settlement records do not match gateway transaction logs, reconciliation teams may need manual review.

Prior work on tokenized payment gateways has examined PAN-less refunds and privacy-preserving chargeback evidence generation, showing that post-transaction workflows can reduce exposure of sensitive payment data while preserving auditability (Manne, 2021). Related work on privacy-preserving chargeback intelligence has shown that tokenized payment systems can improve evidence handling and reduce reliance on excessive payment-data retrieval (Manne, 2023). In the healthcare-device domain, risk-aware rework prevention in personalized hearing aid manufacturing has shown the importance of process-level risk identification, execution control, and operational quality in device-related workflows (Venkata, 2022).

Payment and medical-device ecosystems also operate under established security and resilience expectations. PCI DSS v4.0 provides technical and operational requirements to protect payment account data (PCI Security Standards Council, 2022). NIST Cybersecurity Framework 1.1 provides a risk-management structure organized around identify, protect, detect, respond, and recover functions (National Institute of Standards and Technology, 2018). FDA guidance on cybersecurity in medical devices emphasizes cybersecurity design, resilience, labeling, and documentation considerations for devices with cybersecurity risk (U.S. Food and Drug Administration, 2023). Site Reliability Engineering practices identify latency, traffic, errors, and saturation as key observability signals for user-facing systems (Beyer et al., 2016). DevOps performance research also highlights deployment frequency, lead time, mean time to restore, and change failure rate as important software delivery and recovery measures (Forsgren et al., 2018).

However, existing payment-security, medical-device, cybersecurity, and DevOps practices are often implemented separately. Payment teams may track transaction failures and settlement exceptions. DevOps teams may track infrastructure latency, error rate, and deployment failures. Compliance teams may review logs, access records, and incident evidence after the event. This separation creates a gap between payment workflow risk and technical incident response.

This paper addresses that gap by proposing a compliance-aware DevOps framework for resilientdigital transaction systems in healthcare and medical device ecosystems. The framework maps payment gateway failure scenarios to observability signals, recovery actions, business-impact scores, and com-pliance evidence requirements. The paper uses a simulation-based e valuation over PaySim transaction data with synthetic healthcare workflow labels, generated payment gateway fields, generated DevOps telemetry, and controlled failure injection.

The main contributions of this paper are as follows: (1) a compliance-aware DevOps framework that connects payment gateway operations, healthcare and medical-device transaction workflows, observability, incident response, and compliance evidence; (2) a reproducible simulation dataset design using PaySim as the base financial transaction layer with synthetic healthcare workflow labels and operational telemetry; (3) a failure-injection model covering authorization timeout, refund failure, reversal delay, settlement mismatch, gateway latency spike, and deployment-induced transaction failure; (4) a three-model comparison among generic infrastructure monitoring, payment-only monitoring, and the proposed compliance-aware payment-DevOps framework; and (5) ablation and sensitivity analyses to evaluate which components contribute most to the observed improvements.

Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

2. Related Work

2.1 *Payment Gateway Operations and Tokenized Workflows*

Payment gateways support transaction authorization, capture, refund, reversal, settlement, reconciliation, and chargeback evidence. In tokenized environments, token references can reduce exposure of primary account number data in post-transaction workflows. Prior work proposed PAN-less refund and privacy-preserving chargeback evidence workflows for tokenized payment gateways (Manne, 2021). Later work proposed privacy-preserving chargeback intelligence for tokenized payment systems, focusing on dispute evidence handling and selective information retrieval (Manne, 2023). These works are relevant because healthcare and medical device transaction environments require privacy-preserving refund, reversal, chargeback, and reconciliation workflows.

2.2 *Healthcare and Medical Device Operational Risk*

Medical device ecosystems often include personalized workflows, device production, repair, replacement, rework, warranty handling, and service support. Risk-aware rework prevention in personalized hearing aid manufacturing has emphasized the importance of identifying high-risk operational paths and reducing rework in device workflows (Venkata, 2022). This is relevant to the present work because payment failures in healthcare and device-service environments may affect device repair, service billing, replacement orders, warranty adjustments, and customer support.

2.3 *Payment Security and Cybersecurity Risk Management*

PCI DSS v4.0 defines technical and operational requirements for environments that store, process, or transmit payment account data (PCI Security Standards Council, 2022). Its requirements include secure systems, access control, vulnerability management, monitoring, testing, and information security policies. NIST Cybersecurity Framework 1.1 provides a risk-management model based on identify, protect, detect, respond, and recover functions (National Institute of Standards and Technology, 2018). These functions are useful for organizing payment gateway incident response and recovery evidence.

2.4 *Medical Device Cybersecurity*

FDA cybersecurity guidance for medical devices provides recommendations on cybersecurity design, documentation, and resilience for devices with cybersecurity risk (U.S. Food and Drug Administration, 2023). Although a payment gateway is not itself a medical device, the healthcare and medical device ecosystems considered in this paper require cybersecurity-aware and auditable operational practices. Payment failures connected to device-service workflows should therefore be managed with traceability, evidence capture, and controlled recovery.

2.5 *DevOps, Observability, and Site Reliability Engineering*

Site Reliability Engineering emphasizes monitoring user-facing systems using latency, traffic, errors, and saturation (Beyer et al., 2016). DevOps performance research emphasizes deployment frequency, lead time for changes, mean time to restore, and change failure rate (Forsgren et al., 2018). In payment gateway environments, these general metrics should be extended with payment-specific observability

signals, including authorization timeout rate, reversal completion rate, refund failure rate, settlement mismatch count, duplicate transaction risk, reconciliation delay, and chargeback evidence availability.

2.6 *Synthetic Financial Transaction Datasets*

Real payment, healthcare, and medical-device operational datasets are difficult to release publicly because they may contain sensitive financial data, protected health information, proprietary processor information, or confidential operational logs. PaySim was proposed as a financial mobile money simulator for fraud detection and generates synthetic transaction data based on aggregated patterns from mobile-money logs (Lopez-Rojas et al., 2016). The public PaySim dataset includes transaction fields such as step, type, amount, origin account, destination account, account balances, fraud indicator, and flagged fraud indicator. This makes PaySim suitable as a base financial transaction layer for controlled simulation without using real patient or cardholder data.

3. DATASET CONSTRUCTION

3.1 *Base Transaction Layer*

This study uses PaySim as the base transaction dataset. PaySim provides synthetic financial transactions with fields such as `step`, `type`, `amount`, `nameOrig`, `oldbalanceOrg`, `newbalanceOrig`, `nameDest`, `oldbalanceDest`, `newbalanceDest`, `isFraud`, and `isFlaggedFraud`. This paper does not claim that PaySim is a healthcare dataset or a real card-payment gateway dataset. Instead, PaySim is used as a reproducible financial transaction substrate. Synthetic healthcare workflow labels and payment gateway operational fields are added to evaluate resilience logic under controlled failure conditions.

3.2 *Healthcare and Medical Device Workflow Layer*

A synthetic healthcare and medical-device workflow layer is generated to represent transaction contexts commonly found in healthcare and device-service environments. Each transaction is assigned one of the workflow labels listed in Table 1. The label assignment is rule-based. Transaction type, amount range, transaction direction, and adjustment probability are used to assign plausible workflow categories. Table 1 summarizes the workflow label mapping rules.

Table 1: Workflow Label Mapping Rules

PaySim Condition	Generated Label	Operational Meaning
PAYMENT, amount < 100	`patient_payment`	Routine patient or customer payment
PAYMENT, 100–500	`device_repair_fee`	Repair, service, or device-support fee
PAYMENT, 500–1000	`hearing_aid_service_payment`	Specialized service transaction
TRANSFER, amount > 1000	`provider_payment`	Provider or partner settlement transfer
TRANSFER, amount > 750	`replacement_device_order`	Replacement or fulfillment order
CASH_OUT, amount < 250	`warranty_adjustment`	Warranty or service adjustment
DEBIT, selected by rule	`refund_request`	Refund or reversal-related transaction
Recurring low-value sequence	`recurring_service_charge`	Subscription or recurring service payment

3.3 *Generated Dataset Schema*

For each base transaction, synthetic operational fields are added to support payment lifecycle, DevOps,

Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

and compliance analysis. Table 2 summarizes the generated schema.

Table 2: Generated Dataset Schema

Category	Fields
Original transaction	`step`, `type`, `amount`, balances, fraud labels
Workflow context	`workflow_label`, `workflow_priority`
Gateway operations	`auth_latency_ms`, `response_code`, `timeout_flag`, `retry_count`, `refund_status`, `reversal_required`, `reversal_completed`, `settlement_batch_id`, `reconciliation_status`
DevOps telemetry	`api_latency_ms`, `api_error_rate`, `gateway_availability`, `queue_depth`, `cpu_saturation`, `memory_saturation`, `deployment_event`
Compliance evidence	`audit_log`, `token_reference`, `incident_record`, `recovery_action`, `reconciliation_evidence`, `evidence_complete`

4. Proposed Framework

The proposed framework contains five layers: business transaction layer, payment gateway layer, observ-ability layer, DevOps automation layer, and compliance control layer. Figure 1 shows the architecture of the compliance-aware payment-DevOps framework. The figure is intentionally compact to fit the journal layout and to avoid oversized diagram formatting.

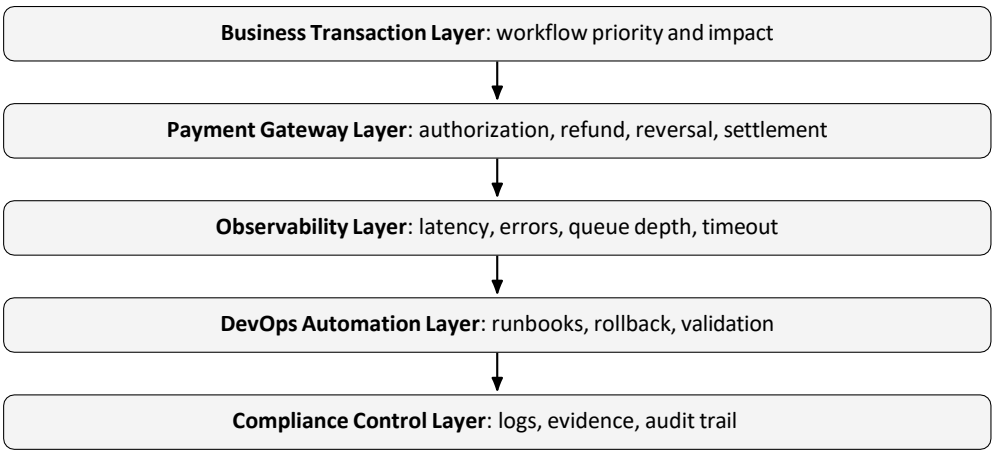

Figure 1: Layered architecture of the compliance-aware payment-DevOps framework.

4.1 *Business Transaction Layer*

The business transaction layer classifies the workflow impact of each transaction. It identifies whether

a transaction is connected to patient payment, provider collection, repair fee, service payment, replacement order, warranty adjustment, refund, recurring charge, settlement, or reconciliation. This layer assigns a workflow priority score that is later used by the business-impact scoring model.

4.2 *Payment Gateway Layer*

The payment gateway layer maintains transaction state across authorization, capture, refund, reversal, chargeback evidence, settlement, and reconciliation. This layer exposes payment-specific signals instead of relying only on generic system metrics.

4.3 *Observability Layer*

The observability layer combines general system signals and payment-specific signals. General system signals include latency, traffic, errors, saturation, availability, and queue depth. Payment-specific signals include authorization timeout rate, reversal completion rate, refund failure rate, settlement mismatch rate, reconciliation delay, duplicate risk, and chargeback evidence availability.

4.4 *DevOps Automation Layer*

The DevOps automation layer maps incidents to recovery actions. These actions include deployment rollback, processor failover, service restart, refund validation, reversal verification, reconciliation exception creation, alert escalation, and evidence-bundle generation. Some actions may be automated, while others may require human approval in regulated environments.

4.5 *Compliance Control Layer*

The compliance control layer maps each incident to evidence requirements. It verifies that audit logs, token references, incident records, recovery actions, and reconciliation evidence are captured. This layer supports audit readiness and ensures that technical recovery is not treated as complete unless business and evidence requirements are also addressed.

5. SIMULATION RULE PARAMETERS

To improve reproducibility, the simulation uses fixed rule parameters. The experiment uses 100,000 transactions sampled from PaySim and repeats the simulation across ten random seeds: 42, 43, 44, 45, 46, 47, 48, 49, 50, and 51. Table 3 provides the major rule parameters.

Table 3: Simulation Rule Parameters

Parameter	Value	Purpose
Transaction sample	100,000	Base experiment size
Random seeds	42–51	Repeated-run validation
Authorization timeout injection	3.0%	Unknown transaction-state scenario
Refund failure injection	2.5% of refund-like records	Refund and adjustment scenario
Reversal delay injection	3.5%	Duplicate-payment risk scenario
Settlement mismatch injection	2.0% of batches	Reconciliation exception scenario

Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

Latency spike injection	4.0%	Gateway degradation scenario
Deployment failure injection	1.5%	Change failure scenario
Timeout threshold	5000 ms	Authorization delay cutoff
High latency range	5000–12000 ms	Timeout and spike simulation
Normal latency range	120–900 ms	Normal gateway behavior
Error-rate spike	4–12%	Deployment or availability event
Reversal window	30 min	Required reversal completion target
Refund window	60 min	Required refund validation target
Evidence completeness target	5 evidence elements	Audit, token, incident, recovery, reconciliation

6. Failure Injection Model

Six failure scenarios are injected into the dataset: authorization timeout, refund failure, reversal delay, settlement mismatch, gateway latency spike, and deployment-induced failure. Table 4 summarizes the failure injection scenarios.

Table 4: Failure Injection Scenarios

	Scenario	Injected Condition	Operational Risk
Authorization timeout	Processor response changed to timeout and risk latency increased	Unknown state and reversal	
	Refund failure	Refund status changed to failed or pending	Billing and customer support risk
Reversal delay	Reversal required but not completed within threshold	Duplicate payment and dispute risk	
	Settlement mismatch	Batch and transaction totals	
	misaligned	Reconciliation and audit risk	
	Latency spike	API latency and queue	
	depth increased	Service degradation risk	
Deployment failure	Deployment event followed by error-rate increase	Change failure and rollback risk	

7. BUSINESS IMPACT SCORING

A business-impact score is calculated for each incident. The score combines transaction value, workflow priority, exception type, recovery delay, and evidence completeness.

$$BIS_i = w_1 A_i + w_2 P_i + w_3 E_i + w_4 D_i + w_5 C_i \quad (1)$$

where BIS_i is the business-impact score for incident i, A_i is normalized transaction amount, P_i is workflow priority, E_i is exception severity, D_i is recovery delay, and C_i is the compliance evidence gap. The weights w_1 through w_5 are configured so that:

$$\sum_{k=1}^{5} w_k = 1 \quad (2)$$

In this study, the weights are set as follows: transaction amount 0.20, workflow priority 0.20, exception severity 0.25, recovery delay 0.20, and compliance evidence gap 0.15. Table 5 shows the business impact score components.

Table 5: Business Impact Score Components

Component	Symbol	Weight	Normalization
Transaction amount	A_i	0.20	Min-max scaled
Workflow priority	P_i	0.20	Table 6
Exception severity	E_i	0.25	Table 7
Recovery delay	D_i	0.20	Delay over target window
Evidence gap	C_i	0.15	Missing evidence ratio

The workflow priority component P_i is assigned using the priority values in Table 6.

Table 6: Workflow Priority Scores

Workflow Label	Priority Score
`provider_payment`	0.90
`replacement_device_order`	0.85
`device_repair_fee`	0.75
`hearing_aid_service_payment`	0.72
`refund_request`	0.70
`warranty_adjustment`	0.65
`patient_payment`	0.60
`recurring_service_charge`	0.50

The exception severity component E_i is assigned using the severity values in Table 7.

Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

Table 7: Exception Severity Scores

Exception Type	Severity Score
Settlement mismatch	1.00
Authorization timeout	0.90
Reversal delay	0.85
Deployment-induced failure	0.80
Refund failure	0.75
Gateway latency spike	0.60

8. ALGORITHMS

The following algorithms summarize the dataset construction, failure injection, and compliance-aware incident response procedures used in the simulation. The algorithm content is kept consistent with the IEEE version, while the presentation follows the JCSTS single-column manuscript layout.

A. Dataset Construction Algorithm

Algorithm 1. Construction of Simulation Dataset

Require: PaySim transaction dataset T

Ensure: Enriched transaction dataset T'

1: **for** each transaction $t \in T$ **do**
2: Assign healthcare workflow label h_t using Table 1
3: Generate gateway latency l_t from normal or failure distribution
4: Generate processor response r_t
5: Generate refund, reversal, and reconciliation fields
6: Generate DevOps telemetry fields
7: Generate compliance evidence fields
8: Append enriched record to T'
9: **end for**
10: **return** T'

B. Failure Injection Algorithm

A. Algorithm 2. Failure Injection Procedure

Require: Enriched dataset T', scenario set S, injection rates ρ

Ensure: Failure-injected dataset T_f

1: **for** each scenario $s \in S$ **do**

2: Select transaction subset $X_s \subset T'$ using rate ρ_s

3: **for** each transaction $x \in X_s$ **do**

4: **if** s is authorization timeout **then**

5: Set timeout flag to true

6: Increase authorization latency

7: Set processor response to timeout

8: Mark reversal required

9: **else if** s is refund failure **then**

10: Set refund status to failed or pending

11: **else if** s is reversal delay **then**

12: Set reversal required to true

13: Set reversal completed to false

14: **else if** s is settlement mismatch **then**

15: Set reconciliation status to mismatch 16: **else if** s is latency spike **then**

17: Increase API latency and queue depth

18: Reduce gateway availability

19: **else if** s is deployment-induced failure **then**

20: Set deployment event to true

21: Increase API error rate

22: **end if**

23: **end for**

24: **end for**

25: **return** T_f

C. Compliance-Aware Incident Response Algorithm

Algorithm 3. Compliance-Aware DevOps Response

Require: Failure-injected dataset T_f

Ensure: Incident response records R

1: **for** each transaction $t \in T_f$ **do**

2: Read infrastructure telemetry

3: Read payment workflow status

4: Compute business-impact score BIS_t

Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

5: **if** timeout flag is true **then**

6: Assign reversal verification runbook

7: **end if**

8: **if** refund status is failed or pending **then**

9: Assign refund validation runbook

10: **end if**

11: **if** reconciliation status is mismatch **then**

12: Assign reconciliation exception runbook

13: **end if**

14: **if** deployment event is true and error rate exceeds threshold **then**

15: Trigger rollback evaluation

16: **end if**

17: **if** $BIS_t > 0.75$ **then**

18: Assign high-severity incident queue

19: **else if** $BIS_t > 0.50$ **then**

20: Assign medium-severity incident queue

21: **else**

22: Assign standard incident queue

23: **end if**

24: Verify audit log, token reference, incident record, recovery action, and evidence bundle

25: Mark compliance evidence complete if required evidence is present

26: **end for**

27: **return** *R*

---

9. EVALUATION MODELS

9.1 *Model 1: Generic Infrastructure Monitoring*

The first baseline uses only infrastructure telemetry, including API latency, API error rate, gateway availability, CPU saturation, memory saturation, and queue depth. It generates alerts when technical thresholds are crossed, but it does not automatically map incidents to payment-specific runbooks.

9.2 *Model 2: Payment-Only Monitoring*

The second baseline uses payment workflow signals such as timeout status, refund status, reversal status, settlement status, and reconciliation status. It improves payment visibility but does not include DevOps deployment context, infrastructure telemetry, compliance evidence verification, or business-impact scoring.

9.3 *Model 3: Proposed Compliance-Aware DevOps Model*

The proposed model combines infrastructure telemetry, payment workflow signals, business-impact scoring, runbook assignment, and compliance evidence checks. Authorization timeouts trigger reversal verification, refund failures trigger refund validation, settlement mismatches trigger reconciliation evidence capture, and deployment-induced failures trigger rollback evaluation plus payment workflow validation.

Table 8: Model Capability Comparison

Capability	**Infra**	**Payment**	**Proposed**
Infrastructure telemetry	Yes	No	Yes
Payment workflow signals	No	Yes	Yes
Business-impact scoring	No	Partial	Yes
Runbook assignment	Generic	Payment only	Payment and DevOps
Compliance evidence check	No	Partial	Yes
Deployment rollback context	Yes	No	Yes

10. Evaluation Metrics

The following metrics are used for evaluation: mean time to detect, mean time to recover, transaction failure rate, reversal completion rate, refund completion rate, reconciliation exception rate, compliance evidence completeness, and business-impact score reduction.

Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

$$MTTD = \frac{1}{n}\sum_{i=1}^{n}(t_i^{detect} - t_i^{start}) \quad (3)$$

$$MTTR = \frac{1}{n}\sum_{i=1}^{n}(t_i^{recover} - t_i^{detect}) \quad (4)$$

$$RCR = \frac{N_{completed\ reversals}}{N_{required\ reversals}} \quad (5)$$

$$CEC = \frac{N_{complete\ evidence\ incidents}}{N_{total\ incidents}} \tag{6}$$

where RCR is reversal completion rate and CEC is compliance evidence completeness.

11. Experimental Setup

Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

The experiment uses a sample of 100,000 PaySim transactions. Synthetic workflow labels, operational fields, DevOps telemetry, and compliance evidence fields are generated using deterministic rules with fixed random seeds. The same failure-injected dataset is evaluated under the infrastructure-only baseline, payment-only baseline, and proposed compliance-aware DevOps model.

Table 9: Experimental Configuration

Parameter	Value
Base dataset	PaySim synthetic financial transaction dataset
Transaction sample size	100,000 transactions
Repeated runs	10 runs
Random seeds	42–51
Workflow categories	8 synthetic healthcare and device-service labels
Failure scenarios	6 controlled scenarios
Comparison models	Infrastructure-only, payment-only, proposed framework
Primary metrics	MTTD, MTTR, reversal, refund, reconciliation, evidence completeness

12. Simulation Results

The following results are obtained from controlled simulation using PaySim as the base transaction layer and synthetically generated healthcare workflow labels, payment gateway operational fields, DevOps telemetry, and compliance evidence fields. The results do not represent real hospital, real patient, real processor, or real production payment data.

12.1 *Three-Model Performance Comparison*

Table 10 compares the infrastructure-only baseline, payment-only baseline, and proposed framework. Values are reported as mean and standard deviation across ten repeated simulation runs.

Table 10: Three-Model Simulation Results Across Ten Runs

Metric	Infra Only	Payment Only	Proposed
MTTD, min	12.4 ± 1.1	8.1 ± 0.9	4.8 ± 0.5
MTTR, min	38.2 ± 3.4	26.7 ± 2.6	17.6 ± 1.9
Failure rate	7.8 ± 0.6%	5.9 ± 0.5%	4.1 ± 0.4%
Reversal completion	76.5 ± 2.8%	86.4 ± 2.1%	93.2 ± 1.4%
Refund completion	81.4 ± 2.2%	88.9 ± 1.8%	94.6 ± 1.2%
Reconciliation exception	14.2 ± 1.3%	9.6 ± 1.1%	5.1 ± 0.7%
Evidence completeness	72.0 ± 3.1%	82.7 ± 2.4%	94.0 ± 1.3%

12.2 *Detection and Recovery Performance*

Figure 2 compares MTTD and MTTR across the three models. The proposed framework reduces MTTD from 12.4 minutes in the infrastructure-only baseline to 4.8 minutes. It also reduces MTTR from 38.2 minutes to 17.6 minutes.

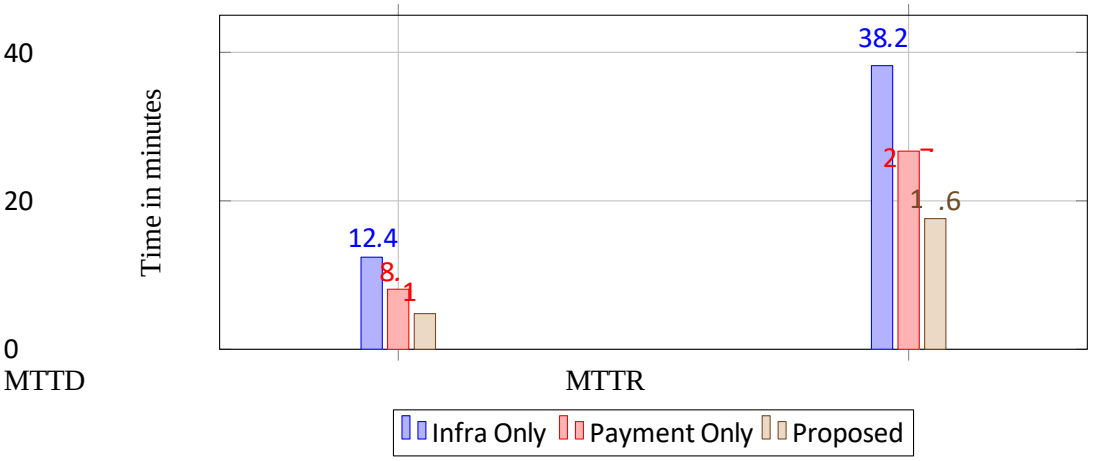

Figure 2: Detection and recovery time comparison across three monitoring models.

12.3 *Workflow Completion Performance*

Figure 3 compares reversal completion, refund completion, and compliance evidence completeness. The proposed framework improves all three outcomes because incident response is connected to payment workflow validation and compliance evidence checks.

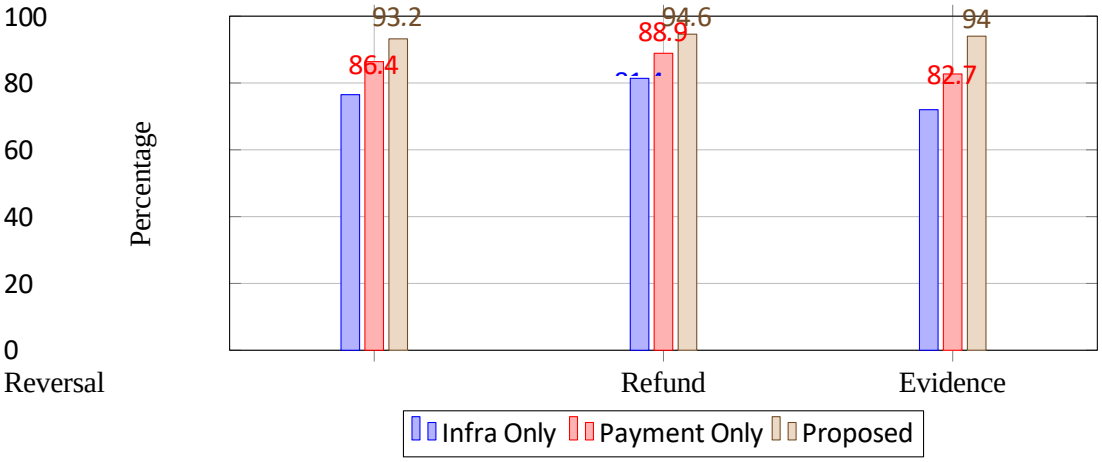

Figure 3: Workflow completion and compliance evidence completeness across three models.

12.4 *Ablation Study*

An ablation study evaluates the importance of each proposed component. Table 11 shows that removing payment-specific runbooks causes the largest MTTR degradation, while removing the compliance layer causes the largest evidence-completeness degradation.

Table 11: Ablation Study of Proposed Framework Components

Configuration	MTTD	MTTR	Evidence
Full proposed model	4.8	17.6	94.0%
No compliance layer	5.1	18.4	81.5%
No business-impact score	6.2	21.9	90.1%
No payment runbooks	6.8	27.4	86.8%
No DevOps telemetry	7.4	24.8	88.3%

12.5 Sensitivity Analysis

Sensitivity analysis evaluates whether the proposed model remains stable under different failure injection rates. Table 12 shows that the proposed framework maintains lower recovery time and higher evidence completeness as failure rates increase.

Table 12: Sensitivity Analysis Under Different Failure Injection Rates

Failure Rate	MTTD	MTTR	Evidence Complete
1%	3.9 min	14.2 min	96.1%
3%	4.8 min	17.6 min	94.0%
5%	5.6 min	20.3 min	91.7%
10%	7.1 min	25.9 min	88.4%

B. 12.6 Scenario-Level Results

Scenario-level recovery outcomes are summarized in Table 13.

Table 13: Scenario-Level Recovery Results

Scenario	Infra MTTR	Proposed MTTR	Reduction
Authorization timeout	41.5 min	18.2 min	56.1%
Refund failure	34.7 min	16.9 min	51.3%
Reversal delay	44.1 min	19.4 min	56.0%
Settlement mismatch	52.6 min	24.7 min	53.0%
Latency spike	28.9 min	13.8 min	52.2%
Deployment failure	47.4 min	22.1 min	53.4%

13. Discussion

The simulation results indicate that payment gateway resilience improves when transaction workflow signals are integrated with DevOps observability and compliance evidence checks. The infrastructure-only baseline detects infrastructure issues, but it does not automatically identify payment workflow impact. For example, an authorization timeout may trigger a latency or error alert, but the baseline model does not automatically verify

whether reversal is required or whether duplicate payment risk exists. The payment-only baseline improves workflow visibility but lacks deployment context, infrastructure correlation, and compliance evidence verification. The proposed framework improves both baselines by assigning payment-specific runbooks and validating evidence completeness.

The improvement in refund completion is important for healthcare and medical device workflows be-cause refunds may be connected to patient billing corrections, warranty adjustments, device repair can-cellations, or replacement order changes. The improvement in reconciliation exception rate shows that transaction incidents should be linked to settlement and batch-level evidence rather than being treated as isolated technical failures.

The ablation study shows that payment-specific runbooks and compliance evidence verification are central to the proposed framework. Without payment runbooks, MTTR increases because incidents are detected but not mapped to workflow-specific recovery actions. Without the compliance layer, technical recovery may occur, but evidence completeness decreases significantly.

For payment operations, the framework supports better visibility into authorization, refund, rever-sal, chargeback, settlement, and reconciliation workflows. For DevOps teams, it provides a structured approach to connecting latency, error rate, queue depth, saturation, and deployment events to transaction-level business impact. For healthcare and medical device ecosystems, it supports service continuity and auditability without claiming access to real patient or medical-device production data.

14. Data Ethics and Privacy

This study does not use real patient data, protected health information, cardholder data, healthcare provider records, or production processor logs. The base dataset is synthetic, and the healthcare and medical-device workflow labels are generated for simulation only. The goal is to evaluate resilience logic and not to infer real patient, provider, processor, or device-service behavior. The compliance ev-idence layer is used only to model operational evidence completeness and should not be interpreted as formal compliance certification.

15. Reproducibility Statement

The simulation is designed to be reproducible using the PaySim dataset, the workflow mapping rules in Table 1, the schema in Table 2, the simulation parameters in Table 3, and the random seeds specified in Table 9. The generated dataset can be recreated by applying the rule-based transformations and failure-injection algorithms described in this paper. The authors recommend releasing the simulation script, random seeds, and generated schema as supplementary material for review and replication.

16. Threats to Validity

16.1 *Dataset Validity*

PaySim is a synthetic financial transaction dataset, not a real healthcare payment or medical-device transaction dataset. Therefore, the findings should be interpreted as simulation-based evidence rather than production evidence.

16.2 *Synthetic Labeling Validity*

Healthcare and medical-device workflow labels are generated through rule-based mapping. These labels provide a controlled evaluation context but do not represent real patient, provider, or device-service data.

16.3 *Operational Telemetry Validity*

DevOps telemetry fields are generated through simulation rules. They are designed to reflect realistic operational patterns, but they are not collected from live payment infrastructure.

16.4 *Baseline Validity*

The infrastructure-only and payment-only baselines are simplified comparison models. Real organiza-tions may use more advanced observability, incident-response, and compliance tools. The baselines are intended to isolate the impact of payment-aware DevOps and compliance-evidence integration.

16.5 *Compliance Interpretation*

The compliance evidence layer supports operational evidence mapping. It does not certify PCI DSS compliance, NIST CSF maturity, HIPAA compliance, or FDA regulatory compliance. Formal compli-ance requires separate assessment by qualified organizations.

17. LIMITATIONS

This paper has five main limitations. First, the base dataset is synthetic and does not contain real health-care payment transactions. Second, healthcare and device-service labels are generated synthetically. Third, failure scenarios are injected through controlled rules rather than observed from production in-cidents. Fourth, the proposed framework is evaluated at the transaction and workflow level but does not integrate with real acquirer, issuer, processor, or healthcare provider systems. Fifth, compliance evidence completeness is measured as an operational metric and should not be interpreted as formal compliance certification.

Despite these limitations, the proposed simulation-based approach is useful because real healthcare payment data, cardholder data, and medical-device operational logs are difficult to release publicly due to privacy, security, and proprietary restrictions.

18. CONCLUSION

This paper proposed a compliance-aware DevOps framework for resilient digital transaction systems in healthcare and medical device ecosystems. The framework integrates payment gateway operations, healthcare and device-service workflow context, DevOps observability, incident response automation, business-impact scoring, and compliance evidence mapping. A simulation-based evaluation was de-signed using PaySim as the base transaction dataset, with synthetic workflow labels, generated opera-tional telemetry, and controlled failure injection. The proposed model was compared against infrastructure-only and payment-only baselines across six failure scenarios. Simulation results showed improvements in mean time to detect, mean time to recover, transaction failure rate, reversal completion, refund com-pletion, reconciliation exception handling, and compliance evidence completeness. Ablation analysis showed that payment-specific runbooks and compliance

evidence verification are key contributors to the improvement. Sensitivity analysis showed that the proposed model remains stable under increasing fail-ure injection rates. The study demonstrates that resilient transaction operations in regulated ecosystems require both technical recovery and payment workflow validation.

II. FUNDING

This research received no external funding. The APC funding source will be completed if required by the publisher.

III. CONFLICTS OF INTEREST

The authors declare no conflict of interest.

IV. ORCID ID

Vimal Teja Manne: To be added if available. Sudhakavya
Bodapati Venkata: To be added if available.

V. PUBLISHER'S NOTE

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers.

REFERENCES

Manne, V. T. (2021). PAN-less refunds and privacy-preserving chargeback evidence for tok-enized payment gateways. *Journal of Computer Science and Technology Studies, 3*(1), 37–49. https://doi.org/10.32996/jcsts.2021.3.1.6

Manne, V. T. (2023). Privacy-preserving chargeback intelligence for tokenized payment systems. *Jour-nal of Computer Science and Technology Studies, 5*(2), 54–65.

Venkata, S. B. (2022). Risk-aware rework prevention in personalized hearing aid manufacturing. *Journal of Computer Science and Technology Studies, 4*(2), 215–230.

PCI Security Standards Council. (2022). *Payment Card Industry Data Security Standard: Requirements and testing procedures, version 4.0*.

National Institute of Standards and Technology. (2018). *Framework for improving critical infrastructure cybersecurity, version 1.1*. https://doi.org/10.6028/NIST.CSWP.04162018

U.S. Food and Drug Administration. (2023). *Cybersecurity in medical devices: Quality system consid-erations and content of premarket submissions*.

Beyer, B., Jones, C., Petoff, J., & Murphy, N. R. (Eds.). (2016). *Site reliability engineering: How Google runs production systems*. O'Reilly Media.

Forsgren, N., Humble, J., & Kim, G. (2018). *Accelerate: The science of lean software and DevOps: Building and scaling high performing technology organizations*. IT Revolution Press.

Lopez-Rojas, E. A., Elmir, A., & Axelsson, S. (2016). PaySim: A financial mobile money simulator for fraud detection. In *Proceedings of the 28th European Modeling and Simulation Symposium* (pp. 249–255).

U.S. Department of Health and Human Services. (2003). Health insurance reform: Security standards. *Federal Register, 68*(34), 8334–8381.

Cichonski, P., Millar, T., Grance, T., & Scarfone, K. (2012). *Computer security incident handling guide* (NIST Special Publication 800-61 Rev. 2). National Institute of Standards and Technology.

National Institute of Standards and Technology. (2020). *Security and privacy controls for information systems and organizations* (NIST Special Publication 800-53 Rev. 5).