



Banking Observability Monitoring System: An Integrated Framework for Real-Time Financial Transaction Analysis and Anomaly Detection

Smitha T¹, Niveditha N M²

¹Department of Data Engineering, GM UNIVERSITY, Davanagere – 577006, Karnataka, India. smithakatarya@gmail.com

²Assistant professor, Department of information Science and Engineering, School of Computer Science and Technology, Faculty of engineering, GM University, Davanagere. nivedithanmcse@gmail.com

ABSTRACT

Banking systems process a large number of transactions every day. Monitoring these transactions helps users understand transaction activity, system performance, and changes in transaction patterns. It can be difficult to monitor large amounts of transaction data manually. Unusual transactions, changes in transaction status, and performance issues may not be easily identified without proper monitoring and analysis. The Banking Observability Monitoring System provides an interactive dashboard to analyze transaction data, display important statistics, visualize transaction patterns, and identify unusual transactions using a machine learning technique called Isolation Forest. The system allows users to upload a transaction dataset containing transaction IDs, account IDs, amounts, statuses, and timestamps. The uploaded data is processed and displayed through charts, summaries, and performance indicators. Isolation Forest is used to detect unusual transactions. The system also records logs, measures execution time, and generates a summary report. The system provides a visual summary of transaction activities, displays transaction trends and status distributions, highlights potentially unusual transactions, and helps users monitor application performance. Actual experimental results, such as the number of unusual transactions detected and processing time, should be included if they have been measured.

Keywords: Observability, Isolation Forest, Fraud Detection, Transaction Telemetry, Data Preprocessing, System Metrics.

1. Introduction

1.1 Background and Motivation

The banking sector has changed greatly with the rapid growth of digital technology and online financial services. Today, many banking activities, such as fund transfers, bill payments, deposits, and account management, are completed through digital platforms. Due to this digital transformation, banks process a very large number of transactions every day. Monitoring these transactions is essential to ensure that the system works correctly and that financial operations remain secure. If banking transactions are not monitored properly, they may lead to problems such as fraud, financial loss, or system failures. Therefore, modern banking systems require reliable monitoring tools that can analyse transaction data and provide clear insights into system performance.

Banking applications are complex because they consist of multiple services, databases, and processes that work together to perform financial operations. When a problem occurs in any part of the system, it may affect the overall banking service. For this reason, organizations need effective monitoring and observability solutions that help administrators understand system behaviour and detect issues quickly. Observability provides better visibility into system operations by collecting and analysing important information such as logs, metrics, and transaction data.

With observability techniques, banks can monitor system health, identify errors, and track transaction activities more efficiently. These techniques also help organizations detect unusual patterns that may indicate suspicious or fraudulent transactions. Instead of reacting only after a problem occurs, observability allows banks to identify issues early and take preventive actions. This improves system reliability and helps ensure that banking services remain stable and secure for users.

1.2 Problem Statement

Modern banking systems process large numbers of digital transactions through websites, mobile applications, and automated services. As transaction volumes increase, it becomes more difficult to monitor every activity continuously. Traditional monitoring approaches may not provide clear visibility into transaction flow and system behavior, making it challenging to identify failures, abnormal transaction patterns, and possible fraudulent activities.

Manual inspection of thousands of records is time-consuming and inefficient. Delays in identifying errors can affect connected services and create operational or financial risks. Therefore, an integrated framework is needed to collect, clean, analyze, and visualize transaction data while highlighting suspicious patterns and reporting useful performance metrics.

1.3 Objectives

- Monitor banking transactions and retain important transaction details.
- Improve performance visibility by collecting logs and latency metrics.
- Use the Isolation Forest machine learning algorithm to identify unusual transaction patterns.
- Present transaction information through understandable charts and dashboards.
- Support quicker decisions by highlighting key metrics and suspicious records.
- Improve system reliability by helping identify errors and operational issues early.

1.4 Research Contributions

- Develops an interactive dashboard for monitoring banking transaction data.
- Integrates the Isolation Forest algorithm to identify suspicious transactions.
- Provides transaction metrics and visualizations for understanding transaction behaviour.
- Includes logging and execution-time monitoring to evaluate system performance.
- Supports better analysis and decision-making through clear transaction insights.

2. Literature Review

The Internet of Things (IoT) generates continuous sensor data, creating challenges related to security, privacy, data integrity, and abnormal behaviour. Existing research on data management, blockchain, and machine learning provides useful approaches to address these challenges. This section reviews existing methods, identifies research gaps, and explains the need for an integrated security framework.

2.1 Existing IoT Data Management Approaches

Traditional IoT systems use centralized servers or cloud platforms to collect and store sensor data. Security methods such as encryption, authentication, and access control help protect the data. However, centralized systems may face server failures, unauthorized access, and data modification risks. Research on dataset quality, reproducibility, and responsible data management [3], [6], [13] highlights the importance of maintaining reliable and properly managed data.

2.2 Blockchain-Based Security for IoT

Blockchain uses cryptographic hashes and linked records to support data integrity and traceability. In IoT systems, sensor data can be stored in an off-chain database while its hash is recorded on the blockchain. Comparing hashes helps identify unauthorized changes to stored data. However, blockchain may introduce additional processing and storage overhead, requiring an efficient design for resource-constrained IoT devices.

2.3 Machine Learning-Based Anomaly Detection

Machine learning can identify unusual sensor readings and abnormal device behaviour automatically. Autoencoders detect anomalies by measuring differences between original and reconstructed data, while Isolation Forest identifies unusual observations. Research on trustworthy machine learning, privacy, and fairness [1], [9], [12] provides relevant background for developing reliable analytical systems. The proposed framework uses machine learning to detect abnormal sensor readings and supports the results with additional security verification.

2.4 Research Gaps

Existing approaches address data management, integrity, and anomaly detection separately. Centralized storage may be vulnerable to data modification, blockchain can introduce performance overhead, and machine learning may produce incorrect anomaly predictions. Therefore, an integrated framework is needed to combine data validation, anomaly detection, integrity verification, and secure storage while considering IoT resource limitations.

2.5 Comparative Analysis of Existing Studies

The selected studies mainly address dataset quality, privacy, fairness, trustworthy machine learning, and performance monitoring. Their relevance to the proposed framework is summarized below.

References	Research focus	Relevance and limitation
[1], [9],[12]	Privacy, fairness, and trustworthy ML	Support reliable machine learning; direct IoT security applicability requires verification.
[3], [6],[13]	Dataset quality, responsibility, and licensing	Highlight reliable data management but do not directly address blockchain integrity.
[5], [10],[15]	Privacy-preserving learning	Provide background on protecting sensitive data; do not directly implement the proposed framework.
[7],[11],[14], [16]	Ethics, privacy, and accountability	Support responsible data handling but do not directly implement IoT security mechanisms.
[4],[8],[17],[19]	Health data, dataset compliance, fairness, and accessibility	Provide related ethical and data management insights but have limited direct relevance to IoT security.
[20]	Distributed tracing and performance optimization	Provides background for performance monitoring but does not directly establish blockchain-based IoT integrity.

3. Proposed Methodology

The proposed system collects data from IoT sensors, checks the data, detects unusual readings, and protects the data using blockchain technology. It uses MQTT to transfer sensor readings, machine learning to detect abnormal values, and a database to store the information. A dashboard displays the readings and security details.

3.1 System Architecture

The system consists of the following components:

1. IoT Sensor Simulation

- A Python program generates sensor readings.
- It generates temperature, humidity, pressure, gas, and vibration values.
- It also generates some unusual readings to test the system.

2. MQTT Communication

- MQTT transfers sensor readings to the backend.
- Mosquitto acts as the MQTT broker.
- The backend receives the readings through the iot/sensors topic.

3. Data Processing

- The system checks the received sensor data.
- It identifies missing values, invalid data, and duplicate messages.
- Valid data is sent for further processing.

4. Machine Learning-Based Anomaly Detection

- Machine learning analyses sensor readings.
- It identifies values that are different from normal readings.
- Unusual readings are marked as anomalies and assigned a risk level.

5. Security Verification

- The system checks messages for possible security problems.
- Message IDs and timestamps can help identify duplicate or replayed messages.
- Authentication checks help verify authorized devices when implemented.

6. Blockchain Ledger

- The system generates a hash for the sensor data.
- The hash is stored in the blockchain ledger.
- Comparing hashes helps identify whether the data has been changed.

7. Off-Chain Storage and SQLite Database

- Sensor readings are stored outside the blockchain.
- SQLite stores sensor details, anomaly results, risk levels, and blockchain information.
- Storing the actual readings separately helps reduce blockchain storage requirements.

8. Dashboard and Monitoring

- Streamlit displays the sensor readings and system results.
- Users can view normal readings, anomalies, risk levels, and blockchain details.
- The dashboard helps users monitor the system easily.

3.2 Data Preprocessing Techniques

The preprocessing pipeline converts raw banking records into structured data suitable for analysis and anomaly detection. It includes data collection, missing-value handling, type conversion, duplicate removal, outlier detection, scaling, feature engineering, filtering, aggregation, categorical encoding, and session-state management.

Framework pipeline: Data Collection → Missing Value Handling → Type Conversion → Deduplication → Isolation Forest Anomaly Detection → Scaling and Feature Engineering → Filtering and Aggregation → Categorical Encoding → Session-State Maintenance.

Table 1 - Data preprocessing pipeline architecture.

Step	Functional Purpose
1. Data collection	Loads CSV or synthetic data into a DataFrame.
2. Missing values	Identifies and removes or imputes null and empty fields.
3. Type conversion	Converts amounts to numeric values, timestamps to datetime, and categories to appropriate types.
4. Duplicate removal	Removes repeated records caused by retries or data errors.
5. Outlier detection	Uses Isolation Forest to label potentially suspicious transactions.
6. Normalization	Scales numeric features using Min-Max scaling or standardization.
7. Feature engineering	Derives hour, day, month, and account transaction frequency.
8. Data filtering	Selects relevant columns and records to reduce processing overhead.
9. Data aggregation	Summarizes transaction counts and totals by account or time period.
10. Categorical encoding	Maps categories such as Success and Failed to numeric values.
11. Session management	Stores cleaned data in session state to avoid repeated preprocessing.

4. Conclusions and Future Work

4.1 Conclusion

The Banking Observability Monitoring System provides a consolidated approach to transaction monitoring by combining data cleaning, interactive visual analytics, Isolation Forest anomaly detection, and operational telemetry. Bringing historical data analysis and generated transaction monitoring into one dashboard can improve visibility, support earlier investigation of suspicious activity, and help administrators understand the health of digital banking services.

4.2 Future Work

Advanced machine learning: explore Autoencoders and LSTMs for more complex patterns.

Enhanced real-time streaming: support continuous ingestion from multiple sources.

Predictive analytics: forecast traffic spikes and possible failure points.

Interactive reporting: add customized reports, exports, and dynamic filters.

Automated notifications: integrate SMS, email, or webhook alerts.

Security and compliance: add encryption, role-based access control, and compliance checks.

Usability optimization: improve workflows using user feedback and usability testing.

Infrastructure optimization: investigate multi-node scaling and parallel processing for very large datasets.

Acknowledgements

The authors express sincere gratitude to the Department of Data Engineering, GM University for supporting this research project.

References

- [1] Kumar, A., & Verma, P. (2023). Real-time transaction monitoring and risk assessment in digital banking. In *Proceedings of the IEEE International Conference on FinTech Systems* (pp. 112–118).
- [2] Miller, D. K. (2023). *Designing Resilient Digital Banking Systems*. IEEE Press/Wiley.
- [3] Tan, E. S. (2023). Scalable logging and telemetry architectures for high-throughput payment gateways. *IEEE Software*, 40(2), 64–72.
- [4] Ahmed, M. S., & Smith, J. R. (2022). Observability patterns in distributed cloud-native financial services. *IEEE Transactions on Service Computing*, 15(4), 2100–2112.
- [5] Johnston, R. B. (2022). Data preprocessing frameworks for financial anomaly detection. *IEEE Access*, 10, 45210–45222.
- [6] Wang, C. H. (2022). Machine learning approaches to enterprise fraud detection: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 34(9), 4301–4318.
- [7] Zhang, L. et al. (2021). Evaluating microservice observability: Metrics, logs, and traces in practice. In *Proceedings of the IEEE International Conference on Software Engineering* (pp. 805–816).
- [8] Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *IEEE 8th International Conference on Data Mining* (pp. 413–422).