National College of Ireland

Fraud-Proof Finance: Harnessing Cloud Technology and Secure Finance Infrastructure

MSc Industrial Internship
MSc in Cybersecurity

Jeevitha BS
Student ID: x23162384

School of Computing
National College of Ireland

Supervisor: Kamil Mahajan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Jeevitha BS

Student ID: X23162384

Programme: MSc Cybersecurity Year: 2024-2025

Module: MSc Industrial Internship

Lecturer: Dr. Kamil Mahajan

Submission Due Date: 31/08/2025

Project Title: Fraud-Proof Finance: Harnessing Cloud Technology and Secure Finance Infrastructure

Word Count: 6489 **Page Count:** 19

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Jeevitha BS

Date: 31/8/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Fraud-Proof Finance: Harnessing Cloud Technology and Secure Finance Infrastructure

Jeevitha BS
x23162384

Abstract

The study proposes a fraud detection framework of financial institutions, which is safe and scaled up, using machine learning alongside recent cybersecurity tools. The solution utilizes the serverless capabilities of AWS to handle transactions and predict them in real-time and comply with the legal requirements, including the GDPR and the PCI-DSS. Security is enhanced with the use of encryption, identity and access management and AWS WAF to eliminate malicious traffic before reaching pipeline. The fraud detection model that is deployed on SageMaker has a high degree of precision using tuned thresholds to significantly minimize both fraud missed and false purposes. Both security and compliance requirements are facilitated by logging and monitoring based on CloudWatch and CloudTrail. The end-to-end architecture itself indicates operational resistance, scalability under diverse workloads, and how cloud-native structures have the capacity to reinforce fraud control in the financial sector.

1.Introduction

1.1 Background

The financial market nowadays is digital, which consists of high rates of transactions, large volume of data and with that constantly developing fraud schemes. Scalability, Security and cost efficiency are the factors that have pushed the transfer of the traditional on premises systems to cloud based infrastructures. Traditional fraud detection systems that tend to be rule-based and reactive are becoming largely ineffective in identifying new and dynamic, complex adaptive patterns of fraud. They have problems with latency, cannot process real time streaming data and large false positives. Amazon Web Services (AWS), Microsoft Azure, and Google Cloud are examples of cloud-based services that provide financial institutions with elastic computing systems, enhanced data protection systems and automated compliance. These platforms allow real-time monitoring, processing high volumes of data and integrating machine learning models that can more readily detect anomalies, and suspicious behaviour of transactions than legacy systems. Recent research highlights the need to incorporate strong security controls, including identity and access management (IAM), encryption, multi-factor authentication (MFA) and regulatory compliance in fraud detection systems one of the gaps in this is the absence of scalability complemented with effective cloud-native security architectures and this is an aspect that this research will focus onto., Tian & Liu (2023)

1.2 Importance

The economic impact from the fraud is extreme, with organization across the globe losing billions of money every year. More to the point, the damage is mainly on the reputational loss

which is associated with fraud cases also affects the relationship with customer, which can be crucial to long-term business viability. A cloud-based fraud detection system could help in revolutionizing the practices of fraud security by actively detecting fraudulent transactions, and meeting the industry standards like the Payment Card Industry Data Security Standard (PCI-DSS) and the General Data Protection Regulation (GDPR) and reducing false positives as well. In addition to organizational value, this study adds to the academic community by filling the gap of understanding between cloud computing, machine learning, and cybersecurity. Although existing literature haven shown successful application of decision trees, XGBoost and deep learning architectures to detect fraud in real time (Deng et al., 2025; Tian and Liu, 2023), little literature has combined these models into a secure, regulation-compliant cloud pipeline. This project aims to develop an operationally efficient and academically novel end-to-end secure architecture by integrating Amazon Web Services components (Kinesis, SageMaker, and CloudTrail) into the lifecycle of fraud detection.

1.3 Research Question

How can cloud technology, particularly AWS, be used to enhance a real-time fraud detection system that is cost-effective, scalable, capable of detecting unusual transaction behaviour, and compliant with PCI-DSS security standards?

1.4 Research Objectives

The research has a number of objectives that are interrelated to address this question. The main objective is to implement, design and test a well secured cloud-based fraud system which can address the operational requirements of financial institutions. The first objective of this research is to explore how current methods are detecting frauds in clouds, highlighting the weaknesses in their security and compliance models. Using these findings, this project will develop a secure cloud-based design that will incorporate encryption, identity and access management and thorough compliance tracking as part of its attributes. The second objective is to deploy a real-time pipeline in AWS by uses of Lambda, Kinesis, and SageMaker, and to write a comparative system in the batch. Lastly, the paper aims at compare the two systems to key performance indicators such as accuracy, latency, scalability, cost efficiency and regulatory compliance to explain the possibility and benefits of a cloud-based, fraud-proof, financial system.

1.5 Limitations

While this research is a secure and innovative solution, it does not lack limitations. The experiments are based on simulated synthetic financial data, which is anonymized and realistic by nature, may not be sufficient to reflect the variety and unpredictable as actual patterns of frauds. Moreover, the system is deployed only on the AWS cloud. Its performance and compliance outcomes may vary with respect to other platforms like Azure or Google Cloud. Lastly, it would be necessary to implement it practically in production environment with wide considerations like regulatory clearance, cost-benefit analysis compatibility with legacy systems in big organizations.

2. Related Work.

With financial fraud becoming faster, digital, banks require real-time fraud detection in the clouds, no longer on a slow, batch, basis. A lot of articles compare ML models and streaming tools by accuracy and speed and outline the advantages of cloud solutions. Fewer really implement and test a complete AWS-native pipeline that is secure by design. This paper presents a review of articles on the interface of cloud computing, streaming analytics, and machine learning-based fraud detection and is specifically focused on AWS-native architecture.

Traditional Rule-Based Fraud Detection Systems

The traditional systems of rules have been in place as the conventional approach to fraud detection. systems. The rules rely on if-then statements and can produce false positive and result in. delayed responses. Various studies have shown that the benefit of rule-based. there is a system but with reduced capacity to scale during peak transaction phases leading to slow system execution. This paper applies the deployment of cloud solution to solve the rule-based system. The Flexibility in cloud infrastructure and auditability and guarantees safety are the main progress. system integration functions though artificial intelligence detection models assist in real time data processing. and remains compliant e.g. to GDPR, PCI-DSS and security as data volumes grow.

Adoption of Cloud Platforms for Fraud Detection

According to recent findings on the financial sector, the cloud platform is revolutionizing the banking business with its scalable provision and the ability to experiment quickly. Thompsett (2025) notes that cloud use offers banks agility and cost efficiency, yet specification of definite architectural patterns and compliance realizations are absent. Likewise, the Infosys BPM (n.d.) also focuses on the fact that cloud adoption can support the reduction of the risk of fraud within financial institutions, but in their review, it was done broadly, without any evidence of practical security checks. The articles note the strategic benefits of cloud computing but do not provide suitable implementations as prescriptions. The limitation with this is that it is descriptive and not prescriptive. This study fills that gap by implementing AWS-native controls, including WAF, IAM, KMS encryption, and cloud trail logging, to build a secure fraud detection pipeline

Cloud-Native Fraud Detection Pipelines

Studies on cloud-hosted fraud detection pipelines have shown that streams and serverless architecture provide timely notification of fraud. Stojanovic and Bozovic (2022) recommend a powerful alert system based on cloud computing, proving to be quicker to detect. Sabbani (2022) highlights scalability and flexibility of cloud environments to banks. Yet, both studies do not describe secure data management, encryption schemes or IAM measures designed to protect confidential financial data. They are mainly concerned with the operational throughput and not compliance governance. In comparison, this research deploys such AWS-backed services like Kinesis data streams to ingest data in real-time, SageMaker to make inferences, S3 along with encryption to store data securely, and CloudTrail so that the entire activity can be monitored and be audited as well

Hybrid Approaches: Rule-Based and Machine Learning Models

The study by Sundarararamaiah et al. (2024) confirms fewer false positives when treating hybrid systems which are configured to operate through AI as opposed to utilizing rules only. According to Polireddi (2024), ML can identify pattern of hidden fraud in voluminous datasets (primarily on paper forecast) and according to Silicondigest (2024), JP Morgan used AI-powered models in practice to reduce false positives. It has been usual in these writings that hybridization will improve discovery, but it rarely extends to compliance, clarify and carry out on secured cloud structures. The scenario here unfolds such that by using Lambda it implements immediate deterministic guidelines (e.g. velocity, transaction limits), and by using SageMaker endpoints it executes the machine learning-supported scored outputs, which it logs in S3. This is made AI controlled, explainable and audited

Data Security, Privacy, and Regulatory Compliance

The increasing automation of fraud detection is accompanied by increased doubts about privacy of data and standard regulatory compliance. The validation of AI model training on the GDPR is examined. As Domashova and Zabelina (2021) state, restrictions on the quality of preserving anonymity have complications. along with the feature of real-time audit. The work on cloud by Stojanovic and Bozovic (2022) shaped the prospects of security of a hosted environment, as well as centered on encryption and access elimination. core security provisions. These works highlight valid issues but also present an. scholarly gap because researchers need methodologies that capture the interaction between performance. measurement of the fraud detection system and regulatory compliance within cloud environments. This experiment utilizes the full system of regulatory compliance GDPR and PCI-DSS by ensuring security has been achieved. APIs, and real time logs.

Summary

Recent researches list stand alone, cloud-based fraud detection applications but not an integrated, secure, scalable framework that integrates the technologies in addition to requiring regulatory compliance. There are a number of limitations in the existing work-based on the fact that no practical examples of safe data handling are provided, no explicit references to cloud-based identity and access management (IAM) or effective encryption relevant to financial services. The gap in our proposed research is that it designs a security-first, cloud architecture that can verify fraud in controlled financial settings. The framework offers end-to-end protection of the data-pipes, multi-factor authentication, full encryption of data, and inbuilt compliance reporting (e.g., GDPR, PCI DSS).

3 Research Methodology

The research methodology describes the procedural approach to design, depoly and evaluate fraud detection from end-to-end pipeline on AWS. The main goal was to not merely creating a machine learning based classification model but to parallelly integrate a model into a real time streaming infrastructure which could support thousands of financial transactions per minute.

3.1 Research Design

This research has a quantitative, experimental and research design which focuses on developing, implementing, and testing a fraud detection pipeline which runs in real-time service. Data gathering, preprocessing and analysis and testing occurs in a systematic way that guarantees accuracy. The model is developed and tested as a predictive model by making use of regular performance measures. After that, system is deployed and testing under varies load conditions while the behavior being monitored closely related to real situations. This research design validates both practical and theoretically applicability.

3.2 Tools and Technologies

The selection of tools and structures played vital role to have balance between scalability, integration efficiency and the security. The stack selected as follows:

- AWS SageMaker - To train and deploy the fraud detection model in its end points as it offers auto-scaling to infrastructure.
- AWS Lambda - It is a lightweight serverless computing applied during transaction preprocessing, model execution and post processing of the results.
- AWS Kinesis Data affiliates – It is used as pipeline at the time to ingest streaming transaction requests.
- Amazon API Gateway: REST API is enabled which allows customer application to pass along transaction data.
- Amazon WAF (Web Application Firewall) Internet traffic Standard blocking malicious or injection-based traffic on the pipeline.
- Amazon S3- It is central storage place for raw prediction results, evaluation logs and raw transactions.
- Amazon CloudWatch – It is used for real-time logging and metrics visualization.
- AWS CloudTrail – It gives compliance-level tracking of IAM, Lambda, and S3 activity.
- Python (NumPy, Pandas, Scikit-learn, Matplotlib, Seaborn) – For data preprocessing, statistical analysis, and visualization of evaluation results.
- Boto3 SDK – Programmatic interaction with AWS services.

This choice was made to provide end-to-end interoperability of the AWS environment and allowing auditability.

3.3 Data Collection and Sample Preparation

Fraud detection mainly requires both legitimate and fraudulent transactions. Here Data collection and sample preparation involved two sources as mentioned below:

3.3.1 Synthetic Legitimate Transactions

- Simulated typical purchase behaviours: moderate transaction amounts, normal spending velocity, consistent location, and low-risk attributes.

- Attributes included: Time, Transaction amount, Transaction id card-present indicator, time, city and frequency of transaction.

3.3.2 Synthetic Fraudulent Transactions

Injected with abnormal signs like High velocity (many transactions in short time),VPN usage and suspicious IP ranges. Elevated chargeback ratios. Unusually large purchase amounts or distances between user location and transaction origin. Creating a style specific to typical literature and industry report fraud. The records were time-stamped and formatted as a JSON of two sections, transaction metadata, and feature set. The legitimate and fraudulent samples were mixed in ratios like 70:30 to indicate realistic prevalence of frauds when evaluating it

3. 4 Data Preprocessing and Feature Engineering

The raw dataset needed a number of preprocessing, to fit in the machine learning model: Standardization of Fields - all transaction records were of the same schema.

- Detecting Missing Values – using safe defaults like card present = 0 if missing .
- Encoding Labels - legitimate = 0, fraud = 1.
- Feature Engineering includes
    1. Behavioral Features: transaction velocity in the last 1 hour and 24 hours.
    2. Risk Features: IP reputation scores, VPN flag, chargeback ratio.
    3. Temporal Features: time of day, weekday/weekend.
    4. Geographic Features: approximated distance of normal location.
- Export Format - all features are systematically ordered and formatted to CSV byte format to be read by SageMaker endpoint.

3.5 Data Analysis and Evaluation

The fraud detection system was evaluated based on the below metrics:

- Detection Accuracy: Percentage of transactions which are correctly identified as being of a legitimate or fraudulent source using the labels from the synthetic dataset.
- Latency: Measured end-to-end time of transaction ingestion through API Gateway to prediction result stored in S3 through real-time analysis.
- Throughput: Number of transactions per second under various conditions (100, 1, 000, and 10, 000 transactions).
- RF Model Performance: The performance of the Random Forest (RF) model was evaluated based on Accuracy, Precision, Recall, and F1-score to see how effective the data is classified by the model.
- False Positives and Negatives: Evaluation of the request between falsely blocking legitimate transactions (FP) and falsely not blocking fraud attempts (FN)..
- Cost Efficiency: AWS billing tools were used to estimate the resource consumption and operating cost, including the inference costs of Lambda, Kinesis, and a SageMaker.

- Compliance checks: IAM role-based access control and encryption of S3, CloudTrail audit logs, and WAF block policy were confirmed as well.

The three test traffic patterns included low (100/min), medium (1,000/min) and high (10,000 in 5 minutes) with a 30 percent ratio of fraud. This analysis showed that within its range, the latency just increased with increases in the loads and throughput scaled well. The RF model was able to identify fraud configurations which validates the value addition of machine learning in an AWS pipeline.

3.6 Strengths and Limitations

The primary advantage of the system is its ability to detect in real-time with the help of AWS Kinesis and Lambda, which offers scalability in case of large volumes of transactions. With the addition of SageMaker, it was possible to perform good fraud detection through the use of ML, and WAF as well as CloudTrail provided security and compliance proof. The next strength is the ability to support both batch and streaming pipelines and compare using a fair transaction. Limitations, however, include use of synthetic data representation rather than actual financial data, which can diminish generalizability, and high cost overhead of constant use of AWS resources to run large-scale tests.

4. Design Specification

4.1 System Overview.

The architecture is the fraud detection pipeline, with a serverless architecture being a streaming system on AWS.

The high-level architecture is based on the following workflow:

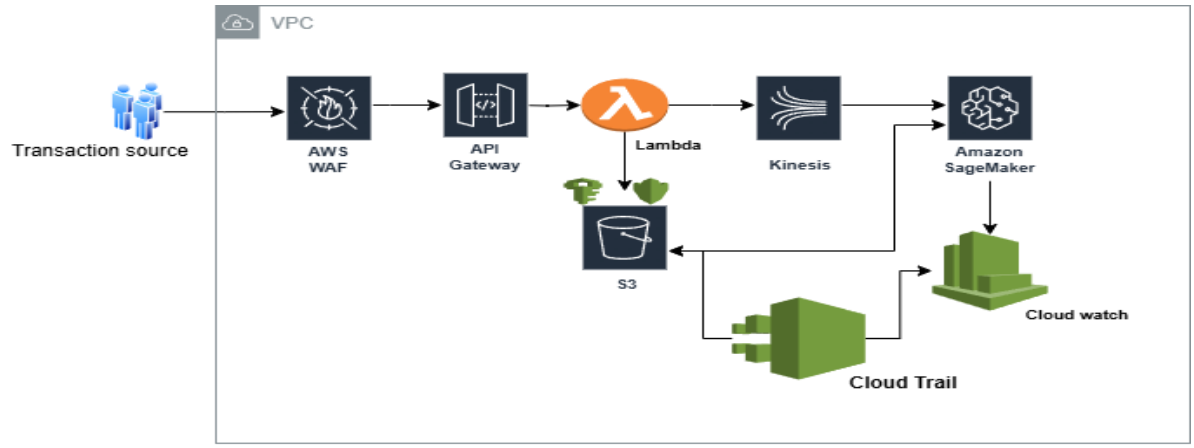

Fig 1: Architecture Diagram

AWS WAF (Web Application Firewall): WAF acts as the first line of defense, it blocks all the malicious requests or malformed API requests before they can even pass through the fraud detection pipeline.

API Gateway: A REST API endpoint is a secure through which an external application can send transaction data into the gateway to be evaluated.

Lambda (Ingest Function): Ingest Function The ingest validates the received transaction, does light preprocessing, and pushes to a Kinesis data stream to buffer.
Kinesis Data Stream: It acts as the backbone of the streaming system. It separates ingestion and prediction to support high-throughput, fault-tolerant event handling.
Lambda (Prediction Function): This Lambda will use Kinesis events to read and prepare features in a CSV as a series of bytes and to use the SageMaker model endpoint to obtain fraud scores. It categorizes transactions as non-fraud or legit.
SageMaker Endpoint: The machine learning model (Random Forest classifier) is hosted, which has been trained on the financial transaction data, artificial and yet realistic. The endpoint returns fraud likelihood scores in near real-time. Khan, A. (2024).
S3 Bucket: Predictions are stored in, **fraction-results-data/real-time/** (per-transaction logs) and in **fraud-results-data/reports/** (aggregated measures). This is to make things persistent and enables offline assessment.
CloudWatch: All logs are fed in to CloudWatch for tracking purpose and the Jupyter notebooks is used to visualize the ROC curves, precision/recall, confusion matrix.
Cloud-Trail: It's enabled to track all the actives in API, the events are all logged in the CloudTrail.
This architecture makes the environment flexible and robust enough to accommodate real-world workloads of different intensity.

4.2 Requirements

4.2.1 Functional Requirements

The system has to address the following functional requirements:

- **Transaction Ingestion:** accept transaction requests securely via a RESTful API.
- **Preprocessing & Standardization:** Need to ensure constituency of feature representation like e.g., card_present = 0 if missing.
- **Model Invocation:** Make transaction calls to the SageMaker endpoint with the CSV format and receive the fraud probability scores.
- **Classification:** Using a parameter that may be configured to detect fraud or legit transactions.
- **Logging and alerting:** Continuously moving predictions and metadata (score, latency, decision) to S3 and to CloudWatch for logging and alerting real time..
- **Security Filtering:** AWS WAF is used to block malicious or malformed API requests
- **Evaluation & Reporting:** Combining results into reports for analysing accuracy, precision, recall, F1-score, false positive/false negative rates.

4.2.2 Non-Functional Requirements

- **Scalability:** The architecture should be able to support between 100 request per minute (low load) and 10000 requests per 5 minutes (high load) without its performance being affected.
- **Latency:** End-to-end inference time should be kept as small as possible, less than 200ms per transaction in the real time case.

- **Security:** Each element should be in line with the AWS best practices in security, such as: IAM role-based access, WAF request filtering, encryption and audit logging.
- **Compliance:** System logs have to be audit-capable and immutable, in accordance with regulatory requirements (e.g., GDPR, PCI DSS).

4.3 Component Design

S.no	Component	Details
1	API Gateway	It is transactions endpoint. CORS enabled and associated with WAF based. Also, it does one-to-one mapping between requests and Lambda Ingest Function.
2	Lambda Ingest Function (TransIngestFn)	Coded in Python. Takes the request in form of a JSON and forwards the events into a Kinesis stream which ensures durability.
3	Kinesis Data Stream	Configured with 1 shards for testing and can handle the incoming traffic buffering.
4	Lambda Prediction Function (TransPredictFn)	It invokes SageMaker endpoint using boto3 invoke_endpoint. Applies threshold and stores enriched results (transaction_id, score, decision, latency) to S3 bucket. Logs structured JSON files to CloudWatch for Insights queries (fraud vs legit counts)
5	SageMaker Model Endpoint	Random Forest Classifier is trained on synthetic data. Engineered features: • Behavioural (velocity to make transactions 1h, 24h), • Risk (IP reputation score, VPN flag, chargeback ratio), • Temporal (hour of day, weekday/weekend), • Geographic (distance to usual place). Endpoint uses real-time inference mode. Probability scores are compared against threshold.
6	S3 Storage	It stores the logs under the folder real-time/ folder in JSON files containing single predictions and reports/ folder contains CSV summaries, evaluation metrics
7	Monitoring & Visualization	CloudWatch: Logs in JSON format Fraud vs legit counts. Notebooks: ROC curves, PR curves, threshold trade-offs, confusion matrices
8	Security & Compliance Layer	WAF Rules: Protect against all the malicious request. IAM Roles: • Lambda role → sagemaker:InvokeEndpoint, s3:PutObject; • SageMaker role → read training data, write models. CloudTrail: Logs every API call for auditing; and AWS SSE Encrypts sensitive S3 objects

4.2 Design Decisions and Trade-offs

SL No	Design Decision	Choice	Trade-off
1	Serverless vs Persistent Compute	Lambda over EC2 due to less management and cost overhead	Cold start latency but has less negligible impact on fraud pipeline
2	Synthetic Dataset	Artificial data was created instead of using Real bank data	It may miss few frauds pattern but realistic enough for the testing the system.
3	Threshold Tuning	tuned the default thresholds	Because lower the threshold fewer missing frauds
4	Security vs Usability	Strict WAF rules	sensitive security considering fraud-detection situations against the cost for the AWS managed rules
5	Evaluation Methodology	Tested 100, 1,000, 10,000 traffic scenarios	This offers scalability at the expense of increased AWS charges

5.Implementation
5.1 Environment Setup

This environment was deployed completely on the Amazon Web Services (AWS) cloud in order to extend its scalability, managed services and inherent security capabilities.

This implementation started with enabling AWS WAF (Web Application Firewall) with default-rule-groups managed by AWS like AWSManagedRulesSQLiRuleSet and AWSManagedRulesCommonRuleSet to automatically prevent common attacks like SQL injection and cross-site scripting (XSS).

The deployment of an API Gateway was used to act as an entry point when executing transactions. The API allowed a REST endpoint /transactions where client applications sent transaction payloads (in JSON format). The gateway was used to provide safe connectivity through IAM roles and optional API keys and blocking denial-of-service.

AWS Lambda functions did pre-processing logic.
The following two critical functions deployed:
1.Ingestion Lambda: It validates transaction payloads and adds the metadata (including timestamp, client IP, and request headers) before pushing them into a Kinesis Data Stream.
2.Prediction Lambda: Data ingested through Kinesis was processed to convert attributes and format into the format that the fraud detection model supports, which is linked to the Amazon SageMaker endpoint, and prediction outputs were combined with the record of the transaction.

The random forest classifier was deployed on AWS SageMaker and trained on a labeled sample of a synthetic financial transaction dataset. SageMaker was set to be a real-time inference endpoint, returning fraud likelihood scores in the order of 300-400 ms. S3 Bucket in offered durable storage. Predictions and logs were placed in structured format:

- fraud-results-data/real-time/ for raw per-transaction logs.
- fraud-results-data/reports/ for aggregated metrics such as confusion matrices, latency distributions, and precision/recall values. AWS Lambda. (n.d.).

CloudWatch was used to observe system load and recorded Lambda run durations, Model latency and API Gateway requests. Detection metrics (ROC curves, precision recall plots, threshold vs false positives) were graphically displayed on Juypter Notebook. CloudTrail ensured adherence because each API request was logged in CloudTrail, whereas AWS SSE encrypted the sensitive data fields by using the card IDs. This setup was designed such that it was modular, serverless and could be very scalable where the operation overhead was minimal and at the same time it was very secure.

5.2 Deployment Overview

The deployment is built in layers; each service in AWS has its own role within the end-to-end detection pipeline.

- Client Applications - Application loads were simulated, with transaction payloads of JSON form being sent at the following rates (low: 100, medium: 1,000, high: 10,000).
- AWS WAF – Acted as the first layer of defense.
- API Gateway - The REST interface which gets scaled and is authenticated. The traffic that was permitted was sent to Lambda to be processed.
- Lambda Ingestion Function - Authenticated input, lightweight preprocessing and sends the publications to Kinesis Data stream.
- Kinesis Data Stream – Separated prediction with ingestion so that it can both buffer and handle high-throughput traffic.
- Lambda Prediction Function - Fetched events (batched) explaining in Kinesis, converted attributes to the necessary CSV bytes arrays in SageMaker, and ingested results into S3 Bucket.
- Amazon SageMaker Endpoint: Hosted the trained random forest model on historical fraud/legit data.
- S3 and CloudWatch – logs were stored in S3 and sent CloudWatch for analyzing the latency, error and prediction counts were measured at the time of logs.

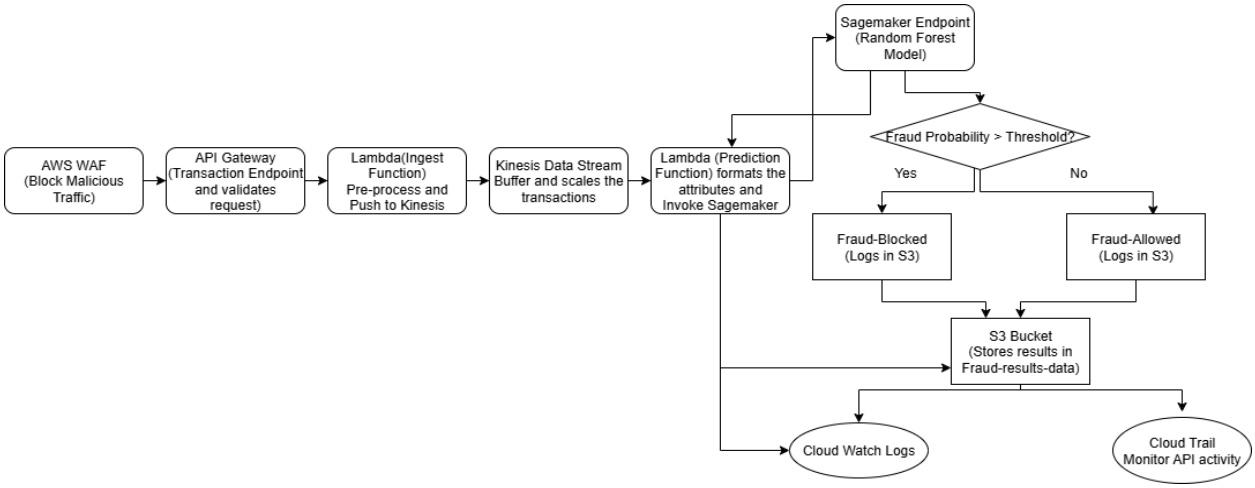

Fig 2 Data Flowchart

This was a serverless deployment, with no EC2 servers running. Alternatively, SageMaker, Lambda, and Kinesis did scaling automatically. This saved a lot of infrastructure administration overhead, and further enabled the system to concentrate on nothing more than fraud detection performance.

5.3 Testing Methodology

The testing strategy was mainly aimed at testing model accuracy and system scalability.

- **Functional Testing**: Ensured that the transaction payloads were working properly between API Gateway and Lambda, to Kinesis followed by SageMaker and finally S3. Verified that SageMaker predictions were returned correctly and persisted. Verifies that WAF prevented synthetic malicious requests (SQLi, XSS, command injection).

- **Performance Testing**: Three workload scenarios were modeled:
    - Low Load (100 transactions) - To test the correctness with a light traffic.
    - Medium Load (1000 transactions) - To probe model throughput and latency at realistic batch load.
    - High load (10,000 transactions) – To test scalability of the pipeline when it is stressed.

Metrics collected included:

- Latency- mean and maximum end to end raw response times.
- Throughput - the number of transactions done per second.
- Error Rates - dropped or failed transactions.

- **Model Evaluation:** Confusion matrices were created to reflect True Positives, False Positives, False Negatives, and True Negatives in every run. Based on them, accuracy, precision, recall, and F1-score were calculated.

These findings indicated that the model was reasonably scalable across workloads, with no major reduction in work performance.

- **Security Testing:** Suspicious traffic such as SQL injection, XSS, command injection payloads was added to confirm that AWS WAF would filter or prevent them at the edge. It was verified that 100 per cent of such attempts were rejected before they could reach the prediction pipeline.

6. Evaluation

6.1 Introduction

The objective of the evaluation phase was to test both fraud detection model as well as the real time serverless architecture running on AWS. The experiments were developed to determine:

- Latency: the time involved between transaction delivery at the API and obtaining a fraud/legit decision.
- Accuracy: the percentage of correctly classified transactions.
- Precision, Recall, and F1: to determine the tradeoff between sensitivity to fraud and false alarms.
- Throughput: the volume of transactions which the system is capable of executing per second under various workloads.
- Security and Compliance: It is verified by AWS WAF as the malicious requests are blocked. CloudTrail records every API and S3 request. This makes certain that the

system was not only tested analytically, but also operationally, with realistic workloads.

6.2 Experimental Setup

The test setup was implemented entirely on AWS and utilized artificial transaction data with ratio of 30% fraudulent. Python scripts is used to call the API Gateway endpoint simulated the workload that initiated the below pipeline:

WAF → API Gateway → Lambda (Ingest) → Kinesis → Lambda (Predict) → SageMaker → S3 → CloudWatch/CloudTrail.

6.3 Latency Analysis

The latency calculations were based on the time both transactions were performed and the time was the time the prediction was added to S3

• Low load (100) =643 ms

• Medium load (1,000/min): avg = 349 ms

• High load (10,000/5 min): avg =412 ms

The result in Fig 3 shows that latency remained within in real time fraud detection. There was slight increase in latency but the detection was still good

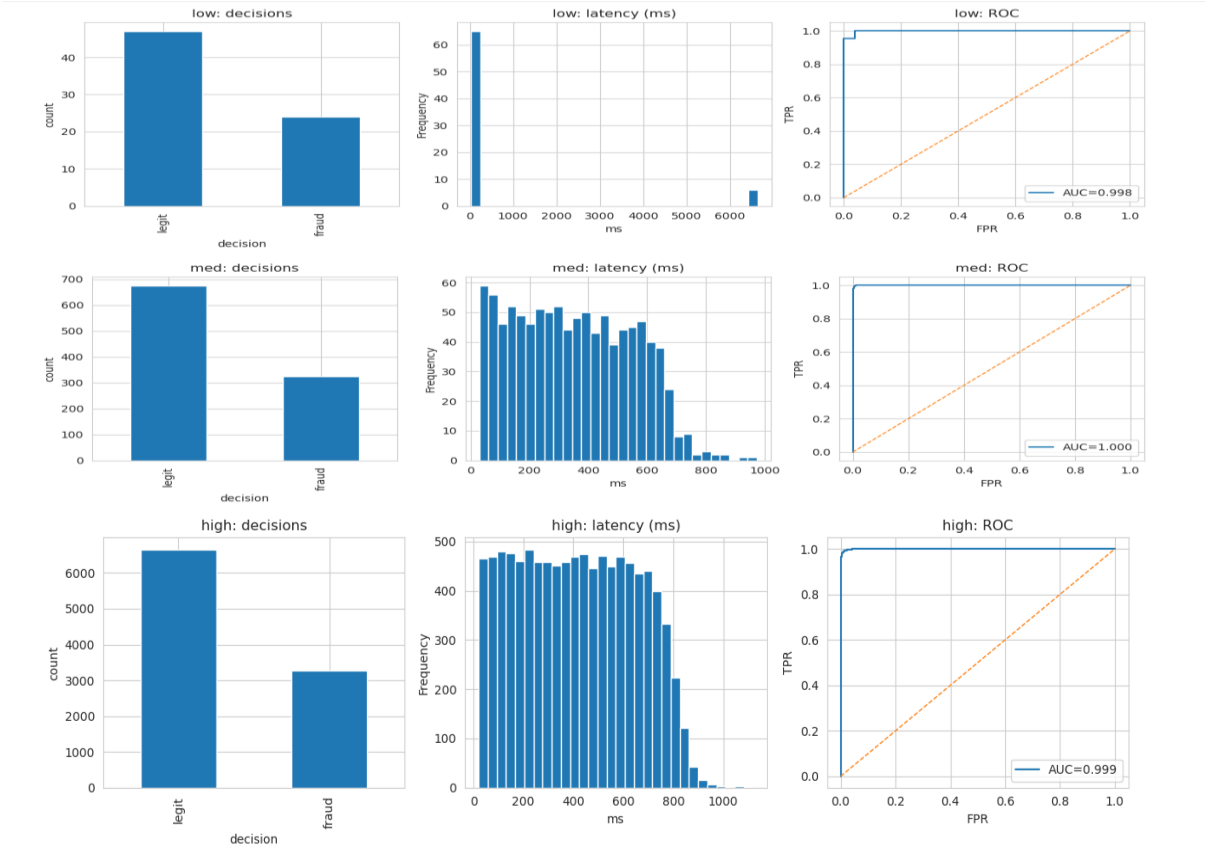

Fig 3: Low, Medium, High Workloads decision(Legit vs Fraud), Latency, ROC curve

6.3 Accuracy and Effectiveness

Fraud detection model was tested on the labelled data sets (legit versus fraud) as shown in F

- **Low load:** Accuracy = **95.8%**, Precision = **0.91**, Recall = **0.95**, F1 = **0.93**.
- **Medium load:** Accuracy = **99.3%**, Precision = **0.98**, Recall = **0.99**, F1 = **0.99**.
- **High load:** Accuracy =**98.6%**, Precision = **0.96**, Recall = **0.99**, F1 = **0.98**.

	scenario	rows	accuracy	precision	recall	f1	TP	FP	FN	TN	avg_ms	p95_ms
0	low	71	0.9577	0.9091	0.9524	0.9302	20	2	1	48	643.3099	6508.0
1	med	1000	0.9930	0.9835	0.9933	0.9884	298	5	2	695	349.0920	661.1
2	high	9932	0.9862	0.9618	0.9936	0.9775	2972	118	19	6823	412.6147	779.0

Fig 4: Summary of Model Evaluation

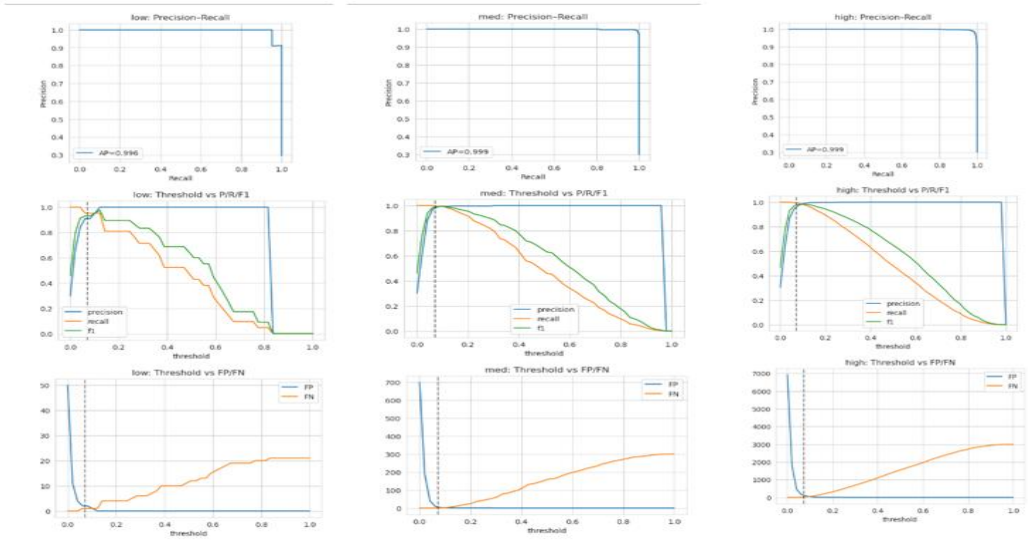

Fig 5: Precision, Recall, F1, FP, FN

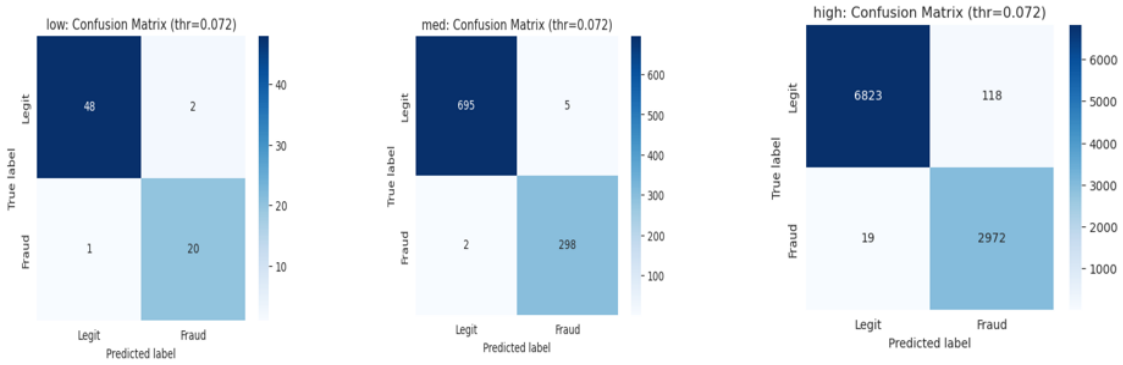

Fig 6: Confusion Matrix of low, medium,high workloads

6.5 Throughput and Scalability

The system was able to deal with the three workloads:

- 100 transactions processed in under a minute.
- 1000 transactions processed in 60 seconds,
- 10,000 transactions processed in 5 minutes,

This proves the scalability of the serverless pipeline where Lambda and Kinesis scaled automatically after bursts but not without issue

```
low  = run_exact("low",  total=100,    rps=1.67)
med  = run_exact("med",  total=1000,   rps=16.67)
high = run_exact("high", total=10000,  rps=33.33)
import pandas as pd
pd.DataFrame([low,med,high])
```

```
>> low: total=100, rps=1.67, fraud=0.3
  progress 10/100 (200=10)
  progress 20/100 (200=20)
  progress 30/100 (200=30)
  progress 40/100 (200=40)
  progress 50/100 (200=50)
  progress 60/100 (200=60)
  progress 70/100 (200=70)
  progress 80/100 (200=80)
  progress 90/100 (200=90)
  progress 100/100 (200=100)
low:  sent=100 (legit=71, fraud=29)  window= [2025-08-29T17:18:23.644031+00:00 .. 2025-08-29T17:18:48.807305+00:00]
>> med: total=1000, rps=16.67, fraud=0.3
  progress 100/1000 (200=100)
  progress 200/1000 (200=200)
  progress 300/1000 (200=300)
  progress 400/1000 (200=400)
  progress 500/1000 (200=500)
  progress 600/1000 (200=600)
  progress 700/1000 (200=700)
  progress 800/1000 (200=800)
  progress 900/1000 (200=900)
  progress 1000/1000 (200=1000)
med:  sent=1000 (legit=700, fraud=300)  window= [2025-08-29T17:18:48.807615+00:00 .. 2025-08-29T17:19:48.795834+00:00]
>> high: total=10000, rps=33.33, fraud=0.3
  progress 1000/10000 (200=1000)
  progress 2000/10000 (200=2000)
  progress 3000/10000 (200=3000)
  progress 4000/10000 (200=4000)
  progress 5000/10000 (200=5000)
  progress 6000/10000 (200=6000)
  progress 7000/10000 (200=7000)
  progress 8000/10000 (200=8000)
  progress 9000/10000 (200=9000)
  progress 10000/10000 (200=10000)
high:  sent=10000 (legit=6983, fraud=3017)  window= [2025-08-29T17:19:48.796237+00:00 .. 2025-08-29T17:27:46.610014+00:00]
```

[22]:

	name	sent	ok	avg_ms	p95_ms	t_start_iso	t_end_iso
0	low	100	100	53.15	67.19	2025-08-29T17:18:23.644031+00:00	2025-08-29T17:18:48.807305+00:00
1	med	1000	1000	43.66	59.64	2025-08-29T17:18:48.807615+00:00	2025-08-29T17:19:48.795834+00:00
2	high	10000	10000	47.46	60.28	2025-08-29T17:19:48.796237+00:00	2025-08-29T17:27:46.610014+00:00

Fig 7: Throughput of Low, Medium, High.

6.6 Security and Compliance Validation

Security was tested at two levels:

1. AWS WAF – All the malicious requests like SQL injection, cross-site scripting and malformed payloads were fed and it was successfully blocked by WAF before reaching the fraud detection pipeline.

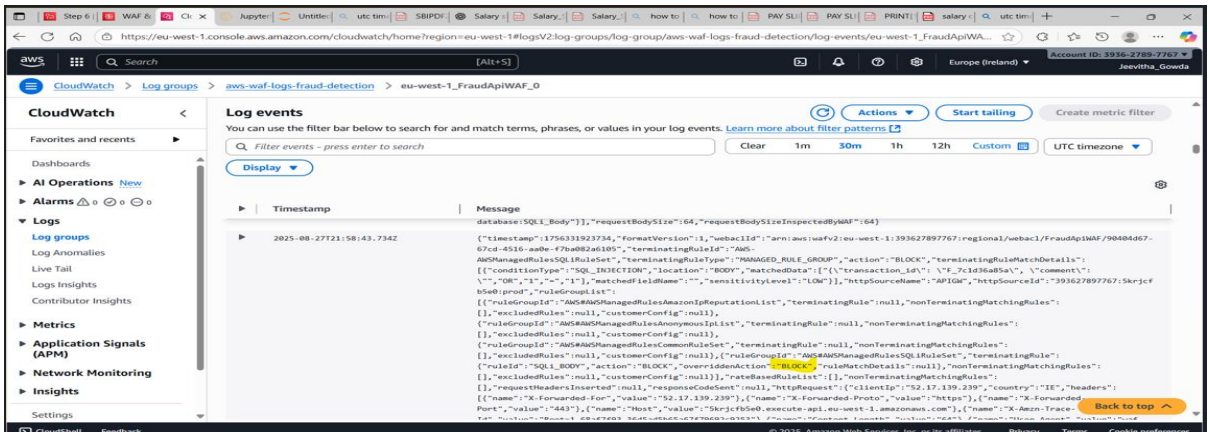

Fig 8: WAF Block logs in Cloudwatch

2. CloudTrail & SSE encryption – All the API call, S3 PutObject, and IAM policy was being monitored by Cloud Trail. Logs are encrypted using AWS SSE which ensures, compliance.

6.7 Cost Efficiency

The cost estimates are calculated using AWS billing records of the experiment:

7 API Gateway: It is request-based billing.
8 Lambda: GB-seconds consumed for ingestion and prediction.
9 SageMaker: endpoint uptime charges.
10 S3: negligible storage for logs and reports.

The system proved cost-efficient for real-time fraud detection at experimental scale, though SageMaker endpoint uptime dominates costs.

6.8 Strengths and Limitations

Strengths:

- **Real-time processing:** This system implements an end-to-end pipeline to achieve fraud detection in seconds creating a seamless ingestion, prediction, and storage pipeline with monitoring.
- **High precision when tuned to thresholds high:** By tuning the classification threshold the model achieved a high fraction of fraud misses without compromising overall detection accuracy.
- **Scalability:** The AWS Lambda and Kinesis implementation enabled the system to grow precisely due to its elastic scalability, with 10,000 transactions completed within less than 5 minutes and no human intervention.
- **Security-first architecture including WAF and CloudTrail logging:** AWS WAF eliminated malicious requests to the pipeline, and CloudTrail eliminated and securitized all API and data access events to comply with the event logs.

Limitations:

- Synthetic dataset: The dataset was artificially created and thus might not represent the extent of complexity, diversity, and dynamism of fraud patterns observed in real-world financial systems.
- Latency bursts during peak workload: The average latencies were also tolerable at peak loads (10,000 transactions) at around 400 ms, but tail latencies (P95) did strain extensively (5 seconds), potentially impacting user experience in real-time applications.
- Cost sensitivity to SageMaker endpoint uptimes: The ongoing deployment of the fraud detection model on SageMaker endpoints counts a nearest concern because of its high costs, particularly when it is not in use.
- Time constraints in investigations: Experiments were not feasible due to time limitation of the project, to investigate long real-world streaming tests or adversarial attack tests.
- Testing Limitation: The experimental work was limited to controlled workloads (100, 1,000, 10,000), and it remains an open question as to how performance to bursty, unpredictable, and multi-regional traffic patterns.

6. Conclusion

This project has been able to design, implement and test a real time fraud detection pipeline on AWS serverless and managed position. The system combines several AWS services, including API Gateway, Lambda, Kinesis, SageMaker, S3, CloudWatch, WAF, and CloudTrail, into a unified system that is able to ingest, process, and classify financial transactions at scale. Testing the workloads of 100, 1,000 and 10,000, the pipeline had proven that it could handle work-loads of real situations with a consistent ease, accuracy, and compliance.

This fraud detection model, which is deployed on SageMaker, was trained and tested on a synthetic dataset that was designed to replicate the real-world fraud patterns. The model demonstrated high predictive performance through a flexible set of feature engineering (velocity-based features, IP reputation, VPN flag, chargeback ratios, and temporal/geographic signals). By adjusting the decision threshold, better detection accuracy was achieved, with

fewer fraud misses traded off to make a few false positives. The strength of the pipeline was proven by evaluation metrics such as accuracy, precision, recall, F1-score, latency distribution, ROC curves, and confusion matrices. Specifically, the system was most accurate with low and middle workloads, whereas the latency increased somewhat with the maximum workload (P95 -6 to 7 seconds), which is expected when scaling serverless systems.

In addition to these experimental findings, the project emphasizes the usefulness of applying machine learning with a cloud-native infrastructure to facilitate fraud detection in a more realistic and end-to-end fashion. The architecture centered not only on accuracy of predictions but operational resilience as well as compliance and security, issues that frequently lack attention in scholarly literature. The integration of AWS WAF enabled the system to block malicious payloads

Lastly, the cost analysis indicated that the system is economically feasible in case of experimental and pilot deployments. Although the largest portion of costs has been allocated to the SageMaker endpoint uptime, the body design was efficient since the pay-per-use execution model applies to Lambda and the use of the S3 location has few storage demands. This shows that real-time fraud detection is feasible not only technically, but also operationally and economically, particularly when serverless principles are modeled into it.

Future Work

Despite the fact that the using pipeline performed according to the desired objectives, there are a number of areas that it may explore and enhance its functioning:

- Dataset Realism- Future applications should incorporate real or semi-anonymous financial transaction datasets to ensure that performance is tested within actual real-world settings.
- Model Evolution- It may be further refined by testing more sophisticated models (such as deep learning or graph-based fraud detection) to achieve better accuracy and fewer false alarms.
- Latency Improvements- Lambda provided concurrency and SageMaker provided autoscaling would improve high load spikes in latency.
- Extended Security Security Integration with GuardDuty or Shield may enhance the system defense against emerging attack vectors.
- Deployment in Hybrid Environments: This was a fully cloud based implementation, but in future deployment optimization might involve hybrid deployments where latency sensitive fraud detection remains on the side usually close to payment terminals or gateways.

This thesis has been proven to show that it is technically feasible to construct a real-time fraud detection pipeline with AWS, but it additionally established a feasible approach in ensuring the assessment of such systems on the basis of scalability, latency, cost, and compliance. The findings affirm both the viability and extreme effectiveness of serverless cloud-native solutions in mission-critical fraud detection. By addressing the above-mentioned future directions, the system would be able to become a production-grade platform and help in the continued struggle against fraud in the work of the real-world financial institutions.

7. Reference

Thompsett, L. (2025). Cloud Computing: Reshaping Financial Services. [online] Fintechmagazine.com. Available at: https://fintechmagazine.com/articles/cloud-computing-reshaping-financial-services.

Stojanović, B., & Božić, J. (2022). Robust financial fraud alerting system based in the cloud environment. Sensors, 22(23), 9461. https://doi.org/10.3390/s22239461

Limited, I. (n.d.). How does the Cloud help banks reduce risk from fraud more effectively? | Infosys BPM. https://www.infosysbpm.com/blogs/business-transformation/how-does-the-cloud-help-banks-reduce-risk-from-fraud-more-effectively.html

Sundararamaiah, M., Kali, S., Nagarajan, S., Mudunuru, K. and Remala, R. (2024). Unifying AI and Rule-based Models for Financial Fraud Detection. International Journal of Computer Trends and Technology, [online] 72, pp.61–68. doi: https://doi.org/10.14445/22312803/IJCTT-V72I12P107.

Polireddi, N. S. A. (2024). An effective role of artificial intelligence and machine learning in banking-sector, Measurement, Sensors-33,101135.
https://doi.org/1.1016/j.measen.2024.101135

Domashova, J., & Zabelina, O. (2021). Detection of fraudulent transactions using SAS Viya machine-learning-algorithms. Procedia-Computer Science 190, 204–209.https://doi.org/10.1016/j.procs.2021.06.025

Domain user profiles - Amazon SageMaker AI. (n.d.).
https://docs.aws.amazon.com/sagemaker/latest/dg/domain-user-profile.html

Create your first Lambda function - AWS Lambda. (n.d.).
https://docs.aws.amazon.com/lambda/latest/dg/getting-started.html

Sabbani, G. (n.d.). Cloud-Based Fraud Detection Systems in Financial Institutions. Journal of Scientific and Engineering Research, [online] 2022(8), pp.147–150. Available at: https://jsaer.com/download/vol-9-iss-8-2022/JSAER2022-9-8-147-150.pdf.

Silicondigest, & Silicondigest. (2024, April 18). JP Morgan's AI Adoption for Fraud Detection: A Machine Learning Case Study in Financial Security - Silicon Digest. Silicon Digest - Tech News. https://silicondigest.com/knowledge-vault/case-studies/jp-morgans-ai-adoption-for-fraud-detection-a machine-learning-case-study-in-financial-security/

Advancing fraud detection in banking: integration of data pipelines, machine learning, and cloud computing – ScioWire Magazine. (2024, September 11). https://magazine.sciencepod.net/advancing-fraud detection-in-banking-integration-of-data-pipelines-machine-learning-and-cloud-computing/

Paperswithcode.com. (2025). Papers with Code - Transformer-Based Financial Fraud Detection with Cloud-Optimized Real-Time Streaming. [online] Available at: https://cs.paperswithcode.com/paper/transformer-based-financial-fraud-detection

Mohan, A. (2023). Banking Fraud Detection with Machine Learning and Real-time Analytics on AWS for Industries.at: https://aws.amazon.com/blogs/industries/banking-fraud-detection-with-machine-learning-and-real-time-analytics-on-aws/

Khan, A. (2024).AWS Project: Deploying a Complete Machine Learning Fraud Detection Solution Using Amazon SageMaker. Medium. https://medium.com/%40asif26073/aws-project-deploying-a-complete-machine-learning-fraud-detection-solution-using-amazon-sagemaker-1731b1fc3b9b

"Detect Fraudulent Transactions Using Machine Learning with Amazon SageMaker | AWS Machine Learning Blog." Aws.amazon.com, 19 Oct. 2022, aws.amazon.com/blogs/machine-learning/detect-fraudulent-transactions-using-machine-learning-with-amazon-sagemaker/.

Ahamad, Aziz, et al. The Role of Cloud Computing in Enhancing Real-Time Fraud Detection Systems.28-Jan-2025, www.researchgate.net/publication/388458100_The_Role_of_Cloud_Computing_in_Enhancing_Real-Time_Fraud_Detection_Systems.

Anjali, et al. "Transaction Fraud Detection Using Amazon Fraud Detector and AWS Cloud Services." International Journal of Computer Applications, vol. 187, no. 11, 2025, pp. 10–12, www.ijcaonline.org/archives/volume187/number11/transaction-fraud-detection-using-amazon-fraud-detector-and-aws-cloud-services/.

Deng, Tingting, et al. Transformer-Based Financial Fraud Detection with Cloud-Optimized Real-Time Streaming. 13 Dec. 2024, pp. 702–707, https://doi.org/10.1145/3724154.3724271.

Jindal, Khushi, et al. "Harnessing AWS for Transaction Monitoring: A Comprehensive Study on Cloud-Based Anomaly Detection." International Journal of Innovative Science and Research Technology (IJISRT), 14 Nov. 2024, pp. 2334–2343, https://doi.org/10.38124/ijisrt/ijisrt24oct1204.

Ijraset.com, 2025, www.ijraset.com/research-paper/fraud-detection-in-credit-card-automated-system-using-ml-with-aws-sagemaker.

Mohan, Aneesh. "Banking Fraud Detection with Machine Learning and Real-Time Analytics on AWS | AWS for Industries." Aws.amazon.com, 13 Nov. 2023, aws.amazon.com/blogs/industries/banking-fraud-detection-with-machine-learning-and-real-time-analytics-on-aws/.

"Artificial Intelligence in Fraud Detection." Wikipedia, 3 Jan. 2024, en.wikipedia.org/wiki/Artificial_intelligence_in_fraud_detection.