# Fintech Use Case: Real-Time Transaction Fraud Detection

Euron

### Fintech Use Case: Real-Time Transaction Fraud Detection

#### Why This Use Case?

In the Fintech industry, fraud detection is critical to preventing financial losses and ensuring customer trust. A real-time fraud detection system can analyze transactions as they happen, leveraging machine learning and stream processing to identify suspicious activities and prevent fraudulent transactions before they occur.

---

# 1. Functional Architecture

This architecture outlines the high-level business and functional requirements for the fraud detection system.

## Actors

- **Customer**: Initiates transactions via banking apps, ATMs, or online portals.
- **Banking System**: Processes and routes transactions.
- **Fraud Detection Engine**: Analyzes transaction data in real-time.
- **Risk Analyst Team**: Reviews flagged transactions for fraud.
- **Security System**: Blocks fraudulent transactions and alerts customers.

## Key Functional Components

1. **Transaction Source (Data Ingestion)**
    - Captures transaction data from multiple sources (mobile banking, credit/debit card swipes, wire transfers).
    - Real-time ingestion using Kafka or AWS Kinesis.
2. **Stream Processing Engine**
    - Consumes transactions in real-time.
    - Enriches transactions with historical customer spending behavior.
    - Applies rule-based and ML-based fraud detection models.
3. **Fraud Detection & Scoring**
    - Assigns a fraud risk score based on predefined rules and ML models.
    - Uses real-time data (location, transaction amount, device fingerprinting).
    - Flags suspicious transactions for review.
4. **Alert & Notification System**
    - Sends real-time alerts to customers for high-risk transactions.
    - Notifies risk analysts for further investigation.
    - Auto-blocks transactions if the risk score exceeds a threshold.
5. **Data Storage & Historical Analysis**
    - Stores transactions for historical fraud analysis.
    - Feeds into ML models for continuous improvement.

6. **Dashboard & Reporting**
    - Provides a real-time view of fraudulent transactions.
    - Allows security teams to analyze trends.

---

# 2. Technical Architecture

This section details the technology stack and workflow of the real-time fraud detection system.

## Technology Stack

- **Data Ingestion**: Apache Kafka / AWS Kinesis
- **Stream Processing**: Apache Flink / Spark Streaming
- **Machine Learning & AI**: Databricks MLflow / TensorFlow / PyTorch
- **Database (OLTP & NoSQL)**: MongoDB / Amazon DynamoDB / PostgreSQL
- **Data Lake (Storage & Batch Processing)**: AWS S3 / Delta Lake / Apache Hudi
- **Analytics & Visualization**: Tableau / Power BI / Grafana
- **Security & Authentication**: OAuth2, JWT, IAM (Identity Access Management)
- **Alerting & Notifications**: Twilio / AWS SNS / Firebase Cloud Messaging

---

## End-to-End Workflow

1. **Transaction Ingestion**
    - Transactions from multiple channels (ATM, mobile app, POS, online banking) are streamed into **Kafka topics**.
    - Kafka partitions transactions based on customer IDs.
2. **Stream Processing & Feature Engineering**
    - **Flink/Spark Streaming** processes incoming transactions.
    - Transactions are enriched with **historical spending patterns**, **device ID**, and **IP geolocation**.
    - Extracted features are passed to the fraud detection model.
3. **Fraud Detection & Risk Scoring**
    - A **Hybrid ML+Rule-Based Model** assigns a fraud risk score (0-100).
    - **High-risk transactions (score >80%) are flagged** and sent for review.
4. **Real-Time Decision Making**
    - If the **score > 90%**, auto-block transaction and notify the customer.
    - If the **score is between 80-90%**, escalate to a risk analyst.
    - If **< 80%**, transaction is processed normally.
5. **Storage & Analytics**
    - **Real-time data is written to MongoDB** for quick access.
    - **Historical data is stored in AWS S3/Delta Lake** for analytics.
    - A **BI Dashboard (Tableau/Power BI)** visualizes fraud trends.
6. **Notification & Customer Alert**

    - High-risk transactions trigger **SMS/email notifications via Twilio/SNS**.
    - Customers can verify transactions through a **self-service app**.

---

# 3. Security & Compliance Considerations

- **PCI-DSS Compliance**: Ensures secure storage of cardholder data.
- **End-to-End Encryption**: TLS 1.3 for data in transit, AES-256 for data at rest.
- **Anomaly Detection**: Uses AI/ML for proactive fraud detection.
- **Access Control**: OAuth2-based authentication for internal and external users.

---

### Benefits of This Architecture

✅ **Real-Time Fraud Detection**: Prevents fraud before transactions are completed.
✅ **Scalable & Resilient**: Kafka and Flink enable high throughput with low latency.
✅ **AI-Powered Insights**: ML models continuously improve fraud detection accuracy.
✅ **Regulatory Compliance**: Secure, auditable, and compliant with financial regulations.