Case Study - Risk Analyst 1 | ClouldWalk

Industry analysis, chargeback resolution, and anti-fraud proposal

João Pedro Tavares
Data Analyst | Candidate for Risk Analyst 1
jjptavares06@gmail.com.com
LinkedIn: linkedin.com/in/joao-pedro-tavares-242189165

Understanding the Payments Industry

1- The customer initiates a card transaction.

⇩

2- Transaction data flows through the payment gateway (if any) to the sub-acquirer and acquirer.

⇩

3- The acquirer sends the request to the card network (e.g., Visa, Mastercard), which routes it to the card issuer (the customer's bank).

⇩

4- The issuer approves or denies the transaction.

⇩

5- The response flows back to the merchant via the same path.

⇩

6- If approved, funds are settled and transferred from issuer to merchant through the intermediaries (with fees applied).

Understanding the Payments Industry

Main Players:

- **Cardholder**: initiates the payment.
- **Merchant**: sells the product/service.
- **Gateway**: routes transaction data (technical layer).
- **Sub-acquirer**: intermediates and aggregates merchant payments.
- **Acquirer**: processes and settles the transaction.
- **Card Network**: connects acquirers to issuers.
- **Issuer**: approves and funds the transaction.

Acquirer vs. Sub-acquirer vs. Payment Gateway

Role	Main function	Financial Risk	Merchant Relationship
Acquirer	Processes Transactions and settles funds	High	Direct
Sub-acquirer	intermediates and aggregates transactions	Moderate	Direct
Gateway	Routes transaction data (technical layer)	None	Direct

Chargebacks, Cancellations, and Fraud

What is a Chargeback?

- A forced reversal of a transaction initiated by the card issuer.
- Usually triggered by disputes such as fraud, non-delivery of product/service, or unauthorized transactions.

Chargeback vs. Cancellation:

- **Chargeback:** initiated by the cardholder's bank after the transaction is processed.
- **Cancellation:** initiated directly by the merchant before settlement, with customer agreement.

Connection to Fraud:

- Fraudulent transactions (especially in card-not-present environments) are a major cause of chargebacks.
- Chargebacks protect customers but pose financial risk to merchants and acquirers.
- Effective fraud prevention reduces chargebacks and losses.

What is Anti-Fraud and How Acquirers Use It

Definition:

- An anti-fraud system is a security layer that analyzes transactions before approval to detect and block fraudulent activity.

How It Works:

- Analyzes transaction data such as IP address, device ID, transaction amount, location, and user history.
- Uses machine learning models, business rules, and risk scoring to classify transactions as “Approve”, “Decline”, or “Review”.

Role of Acquirers:

- Protect the payment ecosystem by minimizing fraud losses.
- Reduce chargebacks and financial risks for merchants and themselves.
- Continuously update and improve fraud detection techniques.

Hands-On Analysis - Detecting Suspicious Behavior

Analysis of provided transactional Data

- Multiple rapid transactions by the same user
- Devices shared by multiple users, indicating potential device sharing or fraud.
- Transactions with unusually high values
- Users and devices with a high rate of fraud chargebacks

(These patterns indicate possible froud and warrant further investigation.)

Additional data to improve fraud detection:

- IP address and geolocation data
- Device fingerprint and browser/app details
- Behavioral data like navigation speed, transaction attempts
- External transaction history and blacklist information
- shared fraud intelligence among acquires

Anti- Fraud Solution Proposal and implementation

Analysis of provided transactional Data

- Real time fraud detection system that evaluates each transaction before approval
- Combines rule based filters and machine learning risk scoring
- Provides a recommendation: “Approve”, “Decline”, or “Review”

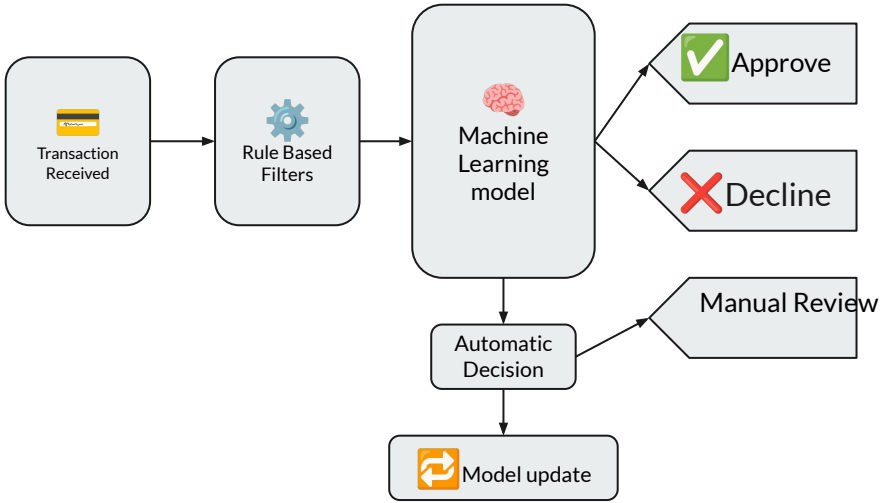
Decision Criteria:

- Reject if multiple transactions occur in a short period by the same user.
- Reject transactions from devices shared by multiple users.
- Reject transactions with values above a defined threshold.
- Flag users or devices with previous fraud chargebacks for manual review.
- Continuously update models with new fraud patterns.

Anti- Fraud Solution Proposal and implementation

Expected benefits:

- Reduce Fraudulent transactions and chargebacks
- Increase merchant and customer trust
- Improve operational efficiency with automated decision making



Continue with the test

```
Windows PowerShell

(.venv) PS C:\Users\helen\OneDrive\Área de Trabalho\Antifraud-API>  uvicorn app.main:app --reload
INFO:     Will watch for changes in these directories: ['C:\\Users\\helen\\OneDrive\\Área de Trabalho\\Antifraud-API']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [12276] using StatReload
INFO:     Started server process [3260]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     127.0.0.1:55317 - "GET / HTTP/1.1" 307 Temporary Redirect
INFO:     127.0.0.1:55317 - "GET /docs HTTP/1.1" 200 OK
INFO:     127.0.0.1:55317 - "GET /openapi.json HTTP/1.1" 200 OK
INFO:     127.0.0.1:55316 - "POST /evaluate HTTP/1.1" 200 OK
INFO:     127.0.0.1:55316 - "POST /evaluate HTTP/1.1" 200 OK
```

Continue with the test

default

POST `/evaluate` Evalute Transaction

Parameters | Cancel | Reset

No parameters

Request body required — application/json

Edit Value | Schema

```
{
  "transaction_id": 0,
  "merchant_id": 0,
  "user_id": 0,
  "card_number": "string",
  "transaction_date": "2025-07-25T01:41:58.365Z",
  "transaction_amount": 0,
  "device_id": 0
}
```

Execute | Clear

Continue with the test

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/evaluate' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "transaction_id": 0,
  "merchant_id": 0,
  "user_id": 0,
  "card_number": "string",
  "transaction_date": "2025-07-25T01:41:58.365Z",
  "transaction_amount": 0,
  "device_id": 0
}'
```

Request URL

```
http://127.0.0.1:8000/evaluate
```

Server response

Code	Details
200	Response body

```
{
  "transaction_id": 0,
  "recommendation": "approve"
}
```

Download

Response headers

```
content-length: 47
content-type: application/json
date: Fri,25 Jul 2025 01:42:00 GMT
server: uvicorn
```

Continue with the test

Responses

Code	Description	Links
200	Successful Response	*No links*

Media type

application/json

Controls `Accept` header.

Example Value | Schema

```
{
  "transaction_id": 0,
  "recommendation": "approve"
}
```

422	Validation Error	*No links*

Media type

application/json

Example Value | Schema

```
{
  "detail": [
    {
      "loc": [
        "string",
        0
      ],
      "msg": "string",
      "type": "string"
    }
  ]
}
```

Continue with the test

Schemas

HTTPValidationError ^ Collapse all object

detail > Expand all array<object>

Recommendation ^ Collapse all object

transaction_id* integer

recommendation* > Expand all string

Transaction ^ Collapse all object

transaction_id* integer

merchant_id* integer

user_id* integer

card_number* string

transaction_date* string date-time

transaction_amount* number

device_id* integer

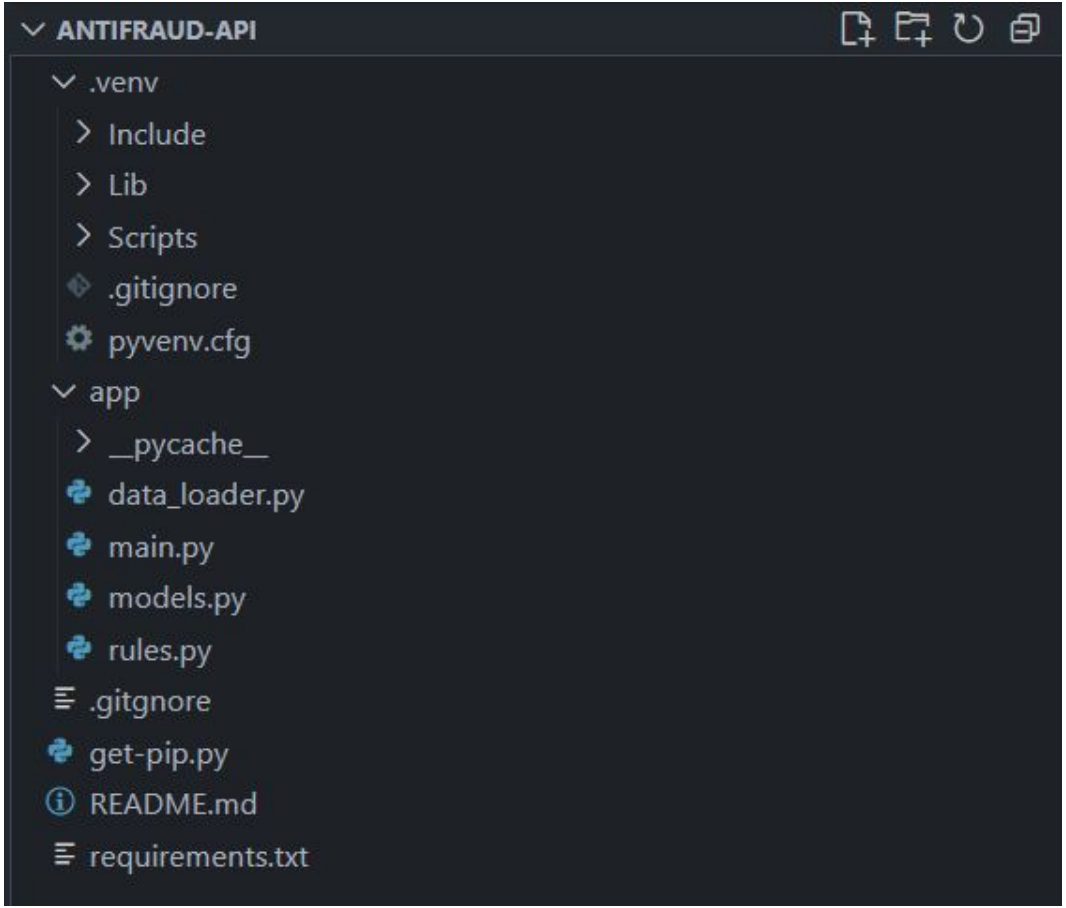
ValidationError ^ Collapse all object

loc* > Expand all array<(string | integer)>

msg* string

type* string

Repository structure



Code structure

data_loader.py

app > data_loader.py > ...

```python
import pandas as pd
import requests
import io

CSV_URL = "https://gist.githubusercontent.com/cloudwalk-tests/76993838e65d7e0f988f40f1b1909c97/raw/295d9f7cb8fd


def load_data():
    response = requests.get(CSV_URL)
    response.raise_for_status()
    df = pd.read_csv(io.StringIO(response.text), parse_dates=["transaction_date"])
    df["has_cbk"] = df["has_cbk"].astype(bool)
    return df
```

Code structure

main.py

app > main.py > root

```python
from fastapi import FastAPI
from fastapi.responses import RedirectResponse
from app.models import Transaction, Recommendation
from app.data_loader import load_data
from app.rules import evaluate

app = FastAPI()
data = load_data()
chargeback_users = set(data[data["has_cbk"]]["user_id"])

@app.get("/", include_in_schema=False)
async def root():
    return RedirectResponse(url="/docs")

@app.post("/evaluate", response_model=Recommendation)
def evalute_transaction(tx: Transaction):
    user_tx = data[data["user_id"] == tx.user_id]
    recommendation = evaluate(tx,user_tx,chargeback_users)
    return{"transaction_id":tx.transaction_id, "recommendation":recommendation}
```

Code structure

models.py

app > models.py > Recommendation

```python
from pydantic import BaseModel
from datetime import datetime
from typing import Literal

class Transaction(BaseModel):
    transaction_id: int
    merchant_id: int
    user_id: int
    card_number: str
    transaction_date: datetime
    transaction_amount: float
    device_id: int

class Recommendation(BaseModel):
    transaction_id: int
    recommendation: Literal["approve", "deny"]
```

Code structure

rules.py

app > rules.py > evaluate

```python
MAX_TRANSACTIONS_PER_HOUR = 3
MAX_DAILY_AMOUNT = 5000.0

def evaluate(tx, user_tx, chargeback_users):
    if tx.user_id in chargeback_users:
        return "deny"

    tx_date = tx.transaction_date
    if tx_date.tzinfo is not None:
        tx_date = tx_date.tz_localize(None) if hasattr(tx_date, 'tz_localize') else tx_date.replace(tzinfo=None)

    one_hour_ago = tx_date - timedelta(hours=1)

    if user_tx["transaction_date"].dt.tz is not None:
        user_tx["transaction_date"] = user_tx["transaction_date"].dt.tz_localize(None)

    recent_tx = user_tx[
        (user_tx["transaction_date"] >= one_hour_ago) &
        (user_tx["transaction_date"] < tx_date)
    ]
    if len(recent_tx) >= MAX_TRANSACTIONS_PER_HOUR:
        return "deny"

    tx_day = tx_date.date()
    daily_total = user_tx[user_tx['transaction_date'].dt.date == tx_day]["transaction_amount"].sum()
    if daily_total + tx.transaction_amount > MAX_DAILY_AMOUNT:
        return "deny"

    return "approve"
```