# Software Engineer - Case

# 1. Introduction

This test is intended for candidates applying to Settlement Analyst positions at CloudWalk. If you get here, we already like you and see you as a good fit with our company. Now, we propose a challenge similar to the ones that we face daily.

The challenges were created to help you build the knowledge base needed to implement the technical assessment. Enjoy!

- The first challenge will help you understand better how the payments industry works.
- The second challenge is a real-world problem.
- The third challenge is an analysis of hypothetical data.

We expect you to understand our role and the challenges facing the financial industry, and to bring data-driven solutions.

# 2. Tasks

## 2.1. Understand the Industry

1. Explain the money flow, the information flow, and the role of the main players in the payment industry.
2. Explain the difference between acquirer, sub-acquirer, and payment gateway, and how the flow explained in the previous question changes for these players.
3. Explain what chargebacks are, how they differ from a cancellation and what is their connection with fraud in the acquiring world.

## 2.2. Solve the problem

A client sends you an email asking for a chargeback status. You check the system and see that we have received his defense documents and sent them to the issuer, but the issuer has not accepted our defense. They claim that the cardholder continued to affirm that she did not receive the product, and our documents were not sufficient to prove otherwise.

You respond to our client informing that the issuer denied the defense, and the next day he emails you back, extremely angry and disappointed, claiming the product was delivered and that this chargeback is not right.

Considering that the chargeback reason is “Product/Service not provided”, what would you do in this situation?

# 3. Get your hands dirty

Attached is a spreadsheet with hypothetical transactional data. Imagine that you are trying to understand if there is any kind of suspicious behavior.

1. Analyze the data provided and present your conclusions.

2. In addition to the spreadsheet data, what other data would you look at to try to find patterns of possible fraud?

3. Considering your conclusions, what could you do to prevent fraud and/or chargebacks?
4. How would you monitor identified patterns?

# 4. Solve the problem

Stop credit card fraud: Implement the concept of a simple anti-fraud.

An Anti-fraud works by receiving information about a transaction and inferring whether it is fraudulent before authorizing it. We work mostly with Ruby and Python, but you can use any programming language that you want.

Please use the data provided in Challenge 2 to test your solution. Consider that transactions with the flag has_cbk = true are transactions with fraud chargebacks.

Your Anti-fraud must have at least: 1 endpoint that receives transaction data and returns a recommendation to “approve/deny” the transaction.

Example payload:

```
{
"transaction_id" : 2342357,
"merchant_id" : 29744,
"user_id" : 97051,
"card_number" : "434505******9116",
"transaction_date" : "2019-11-31T23:16:32.812632",
"transaction_amount" : 373,
"device_id" : 285475
}
```

Example response:

```
{
"transaction_id" : 2342357,
"recommendation" : "approve"
}
```

You are free to determine the methods to approve/deny the transactions, but a few ways to do it are:

- rule-based - you define which cases get approved/denied based on predefined rules;
- score-base - you create a method/model (you could use machine learning models here if you want) to determine the risk score of a transaction and make your decision based on it;
- a combination of both;

Things to watch for:

- Latency
- Security
- Architecture
- Coding style

Antifraud Requirements

- Reject transaction if a user is trying too many transactions in a row;
- Reject transactions above a certain amount in a given period;

- Reject transaction if a user had a chargeback before (note that this information does not come on the payload. The chargeback data is received days after the transaction was approved)

# 5. Deliverables

You are expected to submit a compacted git repository with your answers, any visual representation of the data, and your project.

We hope you have fun, learn, and challenge yourself during this task :)

Feel free to be creative, and good luck!