FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Integration of Fraud Detection Services in Payment Processing Systems

Ana Margarida Ruivo Loureiro

U.PORTO

FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Master in Informatics and Computing Engineering

Supervisor: João Pascoal Faria

Company Supervisor: Simão Belchior

July 29, 2022

Integration of Fraud Detection Services in Payment Processing Systems

Ana Margarida Ruivo Loureiro

Master in Informatics and Computing Engineering

Approved in oral examination by the comitee:

July 29, 2022

Resumo

Atualmente, a indústria de pagamentos encontra-se em crescimento acelerado, com a necessidade de digitalização do mercado a contribuir para este fenómeno. Deste modo, existe uma constante mudança e adaptação às exigências atuais, com as empresas de tecnologia de pagamento a desempenharem um papel mais significativo do que no passado.

As transações de pagamento fraudulentas constituem uma área de crescente preocupação e representam uma percentagem significativa das perdas anuais dos comerciantes. Dada a importância do tema, as plataformas focadas na área de pagamentos devem oferecer ferramentas de deteção de fraudes, denominadas ferramentas de avaliação de risco, de forma a fortalecer a qualidade do serviço oferecido.

No entanto, as soluções existentes no mercado são heterogéneas, e muitas das soluções disponíveis nem sempre são adequadas para todas as transações. Consequentemente, seria interessante analisar uma solução que providencie um sistema configurável de proteção contra fraude. Além disso, este sistema deve ter uma componente de monitorização de resultados, para suportar a tomada de novas decisões de negócio. O objetivo é maximizar a proteção do comerciante à fraude.

Esta dissertação foca-se na melhoria dos sistemas de avaliação de risco destas plataformas, ao desenvolver uma solução capaz de integrar vários fornecedores de ferramentas de risco externo numa única ferramenta, de forma a reduzir a fraude na infraestrutura de pagamento do comerciante. Esta ferramenta apresenta uma interface de fácil uso para o utilizador, que permite configurar a execução do sistema e monitorizar os resultados, conseguindo-se aplicar melhorias na estratégia, mediante a análise dos dados. Muitos fatores contribuem para a complexidade deste problema, como requisitos de tempo, variabilidade de inputs/outputs, condições de execução, regras para agregação de resultados e usabilidade da interface.

A revisão de literatura permitiu obter um valioso conhecimento sobre transações de pagamento, e serviços de deteção de fraude. Além disso, os resultados da solução suportam positivamente o trabalho desenvolvido, já que os parâmetros analisados tiveram todos resultados promissores. Tendo em conta a falta de abordagens semelhantes para a deteção de fraude em sistemas de processamento de pagamentos, este trabalho contém um grande valor inovador, ao documentar e sintetizar a implementação e resultados da solução.

Concluindo, tendo em conta o contínuo crescimento de transações de pagamento, especialmente no setor *online*, e consequente aumento de fraude, o nosso trabalho pode constituir uma primeira abordagem para detetar fraude, imediatamente disponível para todos os comerciantes, que pode beneficiar a indústria de pagamentos.

Keywords: Deteção de Fraude, Indústria de Pagamentos , Avaliação de Risco

Abstract

The payment industry is fast-paced and growing, with the demand for market digitalization contributing to this phenomenon. It is constantly changing and adapting to new requirements, with payment technology companies playing a more significant role.

Fraudulent payment transactions have been a continuously increasing area of concern for a long time and represent a significant percentage of the losses for merchants every year. Therefore, given the importance of this topic, payment-focused platforms should offer fraud detection tools, also referred to as risk assessment tools, to strengthen their service offering.

Nevertheless, the existing market solutions are heterogeneous, and many available solutions are not always adequate for every transaction, so developing a solution that provides a configurable system of fraud detection services would be appealing. Further, it should offer monitoring data to support new business decisions. The goal is to maximize the merchant's resistance to fraud.

This dissertation focus on enhancing existing risk assessment engines by developing an integrated solution, providing different fraud detection services from multiple third parties in a single meta-tool to reduce fraud in the merchant's payment infrastructure. This tool provides the merchant with a user-friendly platform to configure the execution engine based on a system of conditional rules with an applied priority. At the same time, it offers result monitoring to improve the approach chosen. Many factors contributed to the complexity of this solution, such as time requirements, input/outputs variability, execution conditions, rules for results aggregation, and usability.

The literature review provided great insight into payment processing and fraud detection services. The experimental results delivered a positive assertion of the developed work since the parameters studied were all respected with promising outcomes. Furthermore, the usability studies showed that the graphical user interface respects good usability standards.

Given the lack of similar approaches to fraud detection in payment processing, this work contains significant innovative value by documenting and synthesizing the implementation and outcomes of the solution.

To conclude, given the continuous increase of payment transactions, especially in the online sector and the consequent fraud, our work can be a starting point to create a consistent and successful approach to detecting fraud readily available to all types of merchants, which could benefit the payment processing industry.

Keywords: Fraud Detection, Payment Industry, Risk Assessment

Acknowledgements

To professor João Pascoal Faria, my gratitude for the availability to exchange thoughts and the resultant valuable guidance and insights.

To Simão Belchior, for the opportunity and the constant trust and support to always critically explore my ideas.

To Pedro Campos to provide valuable knowledge that made this work possible.

To Cristiana Fernandes, Daniel Esteves, Joana Brandão, Maria Magalhães, Rita Cangueiro and Rita Costa for the time spent helping me validate my user interface, and to all the payment gateway team in SaltPay.

To my friends present through my university years for all the support and the happy moments we created together.

To João, for the constant support and always having the right words in the last five years.

To my family, thank you for providing me with an environment where I could grow and never feel limited to expanding my reality and for everything else.

Ana Margarida Ruivo Loureiro

"If more engineers did not like to put up with technology, they would design things that are a lot simpler to use and not only simpler to use, but more robust."

Radia Perlman

Contents

List of Figures

List of Tables

Abbreviations

1Q	First Quartile
2Q	Second Quartile
3Q	Third Quartile
ATM	Automated Teller Machine
BIN	Bank Identification Number
BNPL	Buy Now Pay Later
CAGR	Compound Annual Growth Rate
FN	False Negative
FP	False Positive
GDPR	General Data Protection Regulation
GUI	Graphical User Interface
IQR	Interquartile Range
PCI DSS	Payment Card Industry Data Security Standard
POS	Point of Sale Terminal
ROI	Return of investment
SEQ	Single Ease Question
SUS	System Usability Scale
TP	True Positive
TN	True Negative
UI	User Interface

Chapter 1

Introduction

1.1 Context

With particular attention to e-commerce, the global growth of digitalisation contributes to the payment's fast-paced, growing industry. In fact, the digital payment market is projected to grow at a compound annual growth rate (CAGR) of 15.2% until 2026, with the Covid-19 outbreak having a considerable impact on this trend [44]. Even with the recent market instability, digital payment continues to be a source of investment.

Furthermore, the payment industry is constantly changing due to new payment methods, mergers and acquisitions, and technology. Focusing on technology companies, they are responsible for the technological advances in this field, and their role in the payments industry is expanding [15].

Additionally, there are multiple features in the payment industry to be explored by companies. However, all commonly handle a critical point in the payment process: fraud detection, a sub-category inside the risk assessment studies. Fraud is a significant problem for merchants, particularly in the online sector. Especially in big merchants, the impact of fraud in the company business and the complexity of detecting it grows even more. Thus, the impact and fraud complexity make the merchant's desire for an adequate system grow even more. According to the 2021 report of Juniper Research, the eCommerce fraud losses reached a global 20 billion dollar expense, corresponding to an annual 18% growth [11] adding to the fact that the previous year had already presented a considerable growth.

SaltPay [18] is an example of a payment company that aggregates several payment solutions inside their payment gateway and will serve as a case study for the development and assessment of our solution.

1.2 Motivation

In the payment market environment, the current risk of fraud is a real threat to merchants and, consequently, a real threat to the success of payment operations.

Annual reports consistently provide insight into a significant percentage of client losses associated with fraud in the payment process of merchants [11]. Given that most transactions pass by a fraud detection service, the results imply lower fraud detection rates than desirable. Therefore, companies must be mindful to focus on improving their fraud risk assessment tools since improvements will work as a differentiating factor from competing companies.

Additionally, it is essential to point out that currently, the most effective fraud solutions work as consulting services and most payment companies have no interest in building it themselves.

Furthermore, several fraud detection services have different points of action and advantages in the market. At the same time, several different transaction types for each merchant should go to different fraud detection services. Thus, combining several existing fraud detection services results in a highly versatile solution that should provide better outcomes and enhance the in-house risk assessment of the fintech companies. Moreover, in this context, Saltpay received a client request to create a system that would aggregate two different third-party fraud detection services to create a personalised risk assessment, reinforcing our basis.

This motivation is ideal for studying new mechanisms and solutions for the existing fraud detection services and constitutes the focus of this thesis.

1.3 Goals and Expected Results

1.3.1 Goals

The main goal of this dissertation is to explore the design, implementation, and validation of a system for integrating fraud detection services in payment processing systems. Although we applied the developed solution to the SaltPay platform, other companies would benefit from using the presented approach.

Our primary focus with this tool is to provide a complete solution to the user. This way, the user should be able to configure, execute and monitor the system. Analysing the monitoring data should allow the user to extract conclusions about improving the system, which can be accomplished by returning to the configuration step.

This way, based on this primary goal, we can divide it into several subgoals:

- **G1** - be able to integrate third-party fraud detection services;
- **G2** - be able to combine the execution of different fraud detection services in the same transaction;
- **G3** - be able to aggregate risk assessment results from different fraud detection third-parties;
- **G4** - allow users to configure the execution process;
- **G5** - allow users to monitor the results of the risk system;
- **G6** - comply with response time constraints;

1.3.2 Expected Results

Following the presented goals, the first outcome of the dissertation is the research and documentation itself. This dissertation will contextualise the work, analyse the problem requirements and delimitate a solution.

Moreover, the developed tool will be another outcome of this work, and we expect it to be a fully functional prototype of our vision. With fully functional, we refer to a prototype that respects the goals defined in the section above.

Finally, it is essential to create a critical analysis of the experimental results, limitations in the development, and how we can implement improvements in the future. Thus, the experimental results and respective conclusions will be another result of our work.

1.4 Document Structure

This document is structured in the chapters detailed as follows:

- Chapter 1 introduces and defines the problem, the context of work, and motivations. After that, we define the goals and expected results.
- Chapter 2 focuses on the literature review of payment processes, risk assessment, fraud detection combination, data warehousing and usability testing.
- Chapter 3 introduces the context in which we will work on our solution and details the requirement analysis for our solution.
- Chapter 4 is centred on defining the conception and implementation of our system, analysing the architectural solution and detailing the solution sections.
- Chapter 5 makes an evaluation of the solution based on the experimental results and usability testing.
- Chapter 6 closes this document with an evaluation of the work developed and considerations of the future work inferred from the explored work.

Chapter 2

Background and State of the Art

This chapter presents the outcomes of the literature revision conducted for this work. We introduce the current scene of the payment industry, explore relevant work and understand the technological needs for the problem. Afterwards, we explore and characterise several existing fraud detection services and study techniques for their combination in the execution process. We explore data warehousing concepts for monitoring capabilities and usability evaluation strategies for user interfaces. Finally, we critically analyse the collected information and explain its pertinence in outlining the solution.

2.1 Payment Processing

The payment industry is a heterogeneous area. Because of the constant innovation and quick evolution, there is a lack of concept formalisation, leading to a deficiency of consistency between the available information as expressed in [49]. For this reason, people can be quickly submerged in conflicting information that confuses them, and the primary processes and concepts of the payment industry can be hard to reach and comprehend. Thus, this section will present the essential concepts and processes to understand this area correctly.

2.1.1 Key Definitions

Credit Card Companies and Networks: The foundation of the payment industry, responsible for the networks that connect all the players. They regulate credit card acceptance and survey transactions between businesses and credit card issuers. Moreover, they are also responsible for the security of the process by creating and enforcing credit card processing rules [51].

Examples: Visa, Mastercard, American Express, Discover

Issuers: Financial institutions *"that provide credit cards to consumers on behalf of the card brands'"* [51]. It is the issuer's responsibility to support financially the transaction made by the consumer. Since this action brings inherent risks, it charges a fee for every transaction. Furthermore, the issuer is responsible for approving or declining a transaction in the authorisation process.

Examples: BNP Paribas, Deutsche Bank, Bankinter

Acquirer: Financial institutions *"that accept and process credit and debit card transactions on behalf of the business"* [51]. Acquirers also referred to as acquiring banks, complete a contract with the businesses defining the conditions offered for the payment processes, agreeing on the settlement interval period, reversals, fees, and more.

Examples: JP Morgan, WorldPay, Barclays

Payment Service Provider: A payment service provider (PSP) manages third-party funds offering merchants multiple payment methods. Typically, a PSP can connect to multiple acquiring banks, card networks, and payment networks, partly removing financial institutions' dependencies from the merchant by eliminating the burden of directly integrating those connections [41].

Examples: Paypal, HiPay, Mollie.

Payment Gateways: A payment gateway represents a technical layer that collects consumer payment information and securely forwards them to the relevant payment service provider or acquirer. The payment gateway validates the consumer's credit card, ensures the funds are available, and sends those funds to the business's account [51, 41].

Examples: Braintree, Bambora, Saltpay Switch Gateway

2.1.2 Payment Processing Steps

This section will explain the two main payment processing steps: authorisation and capture, and settlement. In the end, we will provide context for the payment gateway transaction cycle since it is where we will develop the solution.

2.1.2.1 Authorization and Capture

This process starts with the customer's purchase at a point of sale terminal (POS), which forwards the payment information to the acquirer. PSPs can be used in this step, and payment gateways manage the calls to the PSPs. Afterwards, the acquirer receives the transaction and sends it to the credit card network, which will pass it to the issuer for authorisation. The transaction is approved or declined based on the funds' availability and the cardholder's account status. Finally, the issuer will start the capture process to bill the cardholder, where the customer's bank account balance will be updated. The fund transfer between the issuer and acquirer can occur only afterwards [15]. Figure 2.1 presents the detailed steps of this process.

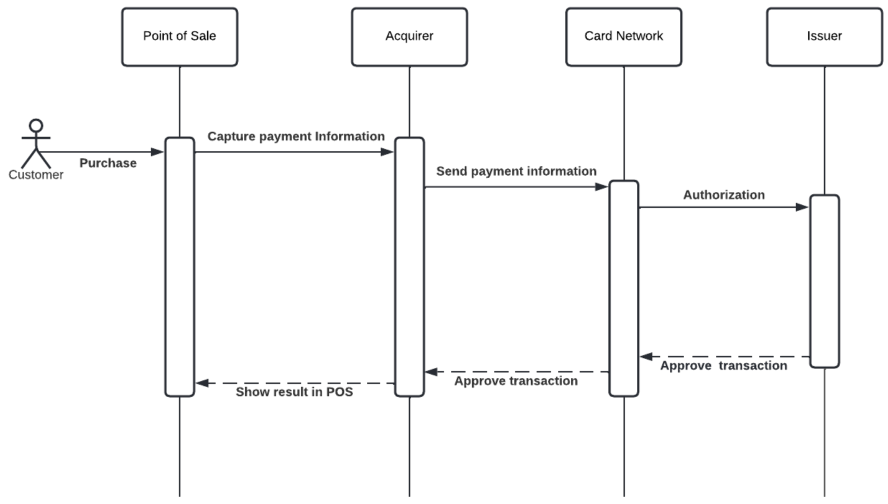


Figure 2.1: UML sequence diagram of a successful authorization process

2.1.2.2 Settlement

The settlement is the process of *"sending funds from the customer's account to the merchant bank."* [7] When the merchant closes the sales period (e.g., at the end of the day), it will transmit all the payment information to its acquiring bank, where a payment processor can be used for support. The acquiring bank will route all the transactions to the issuer via the credit card network. The issuer will transfer the funds to the merchant acquirer charging a fee for this amount. We can find a representation of the described process in Figure 2.2.

2.1.3 Payment Methods

The fast pace in the payment industry caused a complete restructure of the payment institutions and, consequently, the panorama of the payment methods. The data from the European Central Bank reflects these alterations, wherein in 2020, the total number of non-cash payments in the euro area increased by 3.7% to 101.6 billion, and the total value increased by 8.7% to €167.3 trillion. In contrast, the total number of automated teller machines (ATMs) in the euro area decreased by 4.9% to 0.29 million, while the number of point of sale terminals (POS) increased by 4.3% to 12.2 million [21]. Moreover, according to the 2021's World Payments Report, the changes that have been happening in payment methods made them consider 2021 as a transition to a new payments era [20].

Unfortunately, as previously noted, there is a lack of formalisation across the payment industry, reflecting inconsistencies in payment methods' definitions. We will summarise the existing payment methods by providing the definitions we consider in this document.

The traditional payment methods - cash, debit and credit cards, and checks, are morphing into new solutions that gravitate toward digital payments called alternative payment methods. The

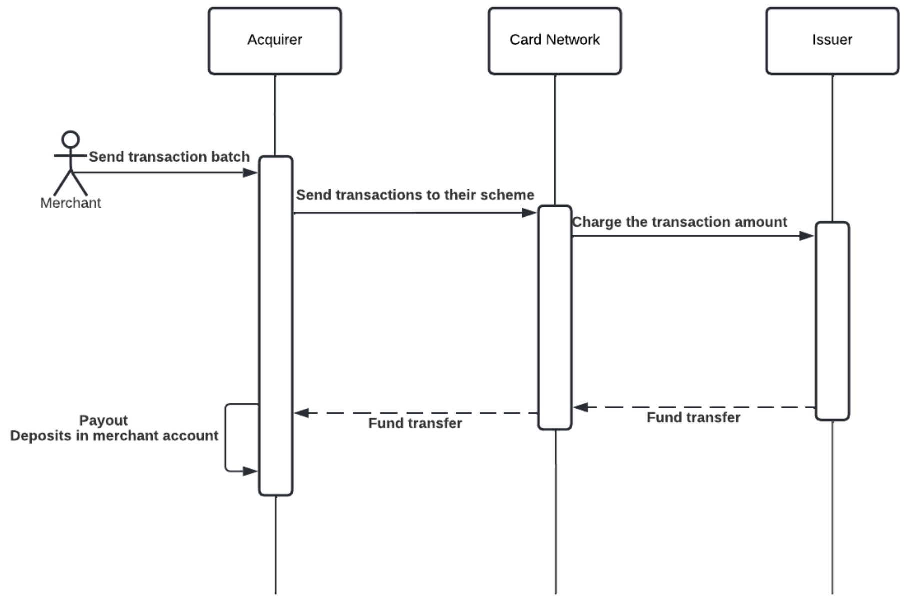

Acquirer
Card Network
Issuer
Merchant
Send transaction batch
Send transactions to their scheme
Charge the transaction amount
Fund transfer
Fund transfer
Payout
Deposits in merchant account

Figure 2.2: UML sequence diagram of the settlement process

rise of e-commerce made the *card not present* approach mainstream, and even the retailers with a physical presence are looking for new ways to provide their customers to pay. Customers choose these methods since they provide less friction and more convenience [1] compared to traditional payment methods. The following methods should be considered in this category: online banking [2], direct debit [12] , digital wallets [6], prepaid cards [17], bank transfers, buy now pay later (BNPL) [33], cryptocurrencies.

Focusing on mobile payments, following eMarketer, between 2020 and 2025, mobile payments are expected to grow with 26.93% of CAGR and encompass many alternative methods such as digital wallets and QR code payments. Further, based on 2021, a total of 25.7% of POS payments were done using mobile wallets [10]. Hence, these results substantiate the growing tendency for alternative payment methods.

2.1.4 Transactions Security

As presented by the 2021 report of IBM [47], all possible evaluation parameters lead to significant growth in the number of data breaches, with the financial sector being the second industry with the most significant average total cost associated with data breaches. These values result from payment systems being critical targets for financially driven attacks, given the nature of the information. Additionally, electronic payment systems such as payment gateways exchange data over an unsecured public network such as the Internet, which presents a significant vulnerability [40, 42]. This section focuses on understanding which security techniques and standards are relevant or needed to respond to our problem.

The Payment Card Industry Data Security Standard (PCI DSS) was defined in 2006 by the major payment card networks to guarantee consumers' credit and debit card data security. It is a set of requirements intended to ensure that all companies that process, store, or transmit credit card information maintain a secure environment to manage PCI security standards and improve account security throughout the transaction process [25].

Thus, it is crucial to analyse the need for PCI DSS compliance in our work. More specifically, we need to ensure that the services transporting sensitive information comply with this standard. Nonetheless, we should be aware of the complexity of the process as described in [16] and the delay that might add to our solution.

Furthermore, we analyse a technique that allows a payment company to reach this compliance. As explained in [19], *"tokenization is a process of replacing sensitive data with non-sensitive data. The payments industry is used to safeguard a card's PAN by replacing it with a unique string of numbers."*. The payment industry actively uses this concept [19, 14] to help maintain a PCI DSS compliant environment. However, this technique should not be applied in this work since third-party fraud detection services require real sensitive parameters and can not be substituted with tokens.

2.1.5 Time Requirements

Time requirements in the payment processing execution are an essential factor to consider. The time response in transaction processing should be small since it influences the operation's success by avoiding the clients abandoning the transaction. In SaltPay's case, they implement a global timeout of the 30s and a 5s timeout in the risk assessment service. SaltPay considers a reasonable time response of around 1 second to provide a good customer experience.

A risk assessment tool from Feedzai declares that it can have a response time of fewer than three milliseconds [28]. However, since we could not find more information about the time response from other fraud detection services, this time value might diverge.

2.2 Risk Assessment

Risk assessment is a defined business process both in literature [43] and industry standards [9, 8, 23]. As detailed in ISO:31000 [9], *"risk assessment is the overall process of risk identification, risk analysis, and risk evaluation"* and has important applications in several fields such as disaster management, security, medicine, and forensics. In our context, risk assessment refers to a critical step in a financial transaction's life cycle to assess the transaction's veracity and possibly affect its completion.

In other words, the primary purpose of fraud detection services is to analyse the risk of fraud in each processed transaction, and the merchant usually uses the resultant risk assessment analysis to apply some action to the transaction. Following the example in Figure 2.3, if the transaction is labelled as risk, it would be blocked while the others accepted. For example, the customer's IP could trigger the signalisation of a high threat. Given this outcome, the merchant rejects it.

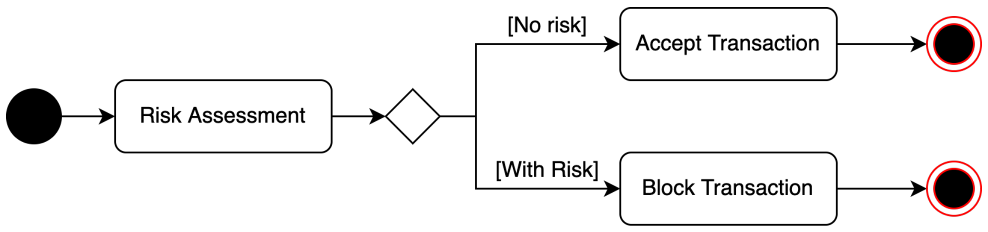


Figure 2.3: UML activity diagram of a high-level view of the risk assessment process for a payment transaction

2.2.1 Quality of Risk Assessment

It is essential to understand how to evaluate a risk assessment service's quality so we can capture its state and compare it with other solutions. Hence, with better mechanisms for improvement, we can captivate more clients.

According to [36] we understand that risk assessment falls into a classification problem since it wants to predict the risk classification of fraud for a given transaction based on multi-dimensional data associated with the transaction metadata. Hence, to evaluate the performance of a risk assessment tool, we focus on existing concepts used in classification problems.

Analysing the values of false positives (FP), false negatives (FN), true positives (TP), and true negatives (TN) in a confusion matrix will help us analyse the performance of the risk assessment tools. Furthermore, precision and recall are performance metrics extracted from the confusion matrix and should also be considered for a more profound analysis of results [3].

In summary, the risk assessment should focus on the following characteristics:

- High percentages of detected fraudulent transactions (TP)
- Low percentages of wrongly classified fraudulent transactions (FP) and undetected fraudulent transactions (FN)

While the consequence of undetected fraudulent transactions is evident, how we handle false positives can influence customer friction, and consequently, it can reflect the abandonment of the negotiation. Mutually influential fraud detection and customer experience should not represent conflicting interests, and we should design the risk assessment tool by minimising its impact on customer friction. This way, analysing the influence of false positives in customer friction allows us to apprehend that a strategy that considers fraud detection based on the worst outcome analysis might not be the best option for our work.

2.2.2 Existing Fraud Detection Services

We performed an extensive analysis of the existing fraud detection services, demonstrating characteristics that allow us to understand the variability and configurability of the several existing solutions in the market and consequently outline the best approach for this work.

All the presented services in the Table 2.1 are available via API, and all the inputs and outputs follow the JSON format. We chose services focused on "smart" data analyses using techniques such as machine learning since we want to focus on available services that will offer a better result than the existing service in SaltPay. We also represent the Saltpay service for comparison purposes. Another selection criterion was the availability of documentation.

We can see equivalent fields in available services, starting with the input analysis. Going through their meaning:

- **Client**: information about the client who started the transaction
- **Payment Method** information about the payment method associated with the transaction
- **Transaction**: transaction characteristics without any category specification
- **Order**: e-commerce related; information about the transaction shopping order, e.g. online shopping in a clothing shop
- **Device**: information of the device used to process or initiate the transaction
- **Risk Profile**: information related to the risk profile of the client associated with the transaction
- **Traveling**: relevant for the travelling sector; travelling information associated with the transaction, e.g. airline ticket
- **Historical Data**: historically collected information about related transactions, e.g. the historical data of a specific client

Table 2.1: Summary of the input fields categories of existing fraud detection services in the market

Fraud Detection Service Name	Input Fields							
	Client	Payment Method	Transaction	Order	Device	Risk Profile	Traveling	Historical Data
Cybersource	X	X		X	X	X	X	X
Fraugster		X		X				
Ravelin		X		X	X			
SaltPay		X	X		X			
MaxMind	X	X		X	X			
Radar	X	X		X				
Adyen		X		X	X	X	X	X
Seon	X	X		X	X		X	
Fraudio	X	X	X		X			X

For example, MaxMind and Cybersource have inputs for the client information. However, the correspondence between fields does not include all the analysed services. This way, we can see that the payment method information is the only field common to all the services, even though many present the device, order and client information fields.

In terms of outputs representing the risk of fraud calculated by the fraud detection analysis, we focused on understanding the types of scales available in Table 2.2. This way, we can see that most resultant scales are probabilistic (continuous quantitative scale) or a qualitative ordinal scale represented by risk levels of actions. For example, Ravelin has three levels of risk: ALLOW - considered genuine order and should proceed - REVIEW - should take extra validation steps to ensure transaction validity - and PREVENT - the transaction is suspected to be fraudulent and should be blocked. In addition, although most risk analyses represent a global scale, some outliers are analysed and presented by singular parameters. An example of a global score is Fraugster, with a 0 to 1 system score representing the fraud probability associated with a transaction.

Table 2.2: Characterisation of the output fields of existing fraud detection services in the market

Risk Assessment Tool Name	Output Data Types			Output Field
	Quantitative	Qualitative		
	Continuous	Ordinal	Nominal	
Cybersource		X	X	global score; individual risk categories
Fraugster	X			global score
Ravelin		X		actions
SaltPay		X		actions
MaxMind	X			score by individual parameters
Radar		X		actions
Adyen	X			global score and individual score
Seon	X			global score
Fraudio	X	X		global score, levels

In summary, each service has a specific set of inputs and outputs in its calls that complies only with the fraud service's internal formats, creating variability in the integration process. This way, the system needs to assess those differences in the integration task.

We conducted a market analysis to comprehend if the services were split by market usage. We defined four main categories:

- Payments refer to payment service providers.
- Services refer to the selling of intangible products.
- E-commerce refers to the typical marketplace of buying and selling products.
- We also considered the airline industry a separate category since it has specific metadata associated with more than one fraud detection service.

As we can see in Table 2.3, CyberSource is dedicated to the airline market, although it presents itself as an across-the-board solution, while Fraudio focuses on payment companies. Furthermore, we found that many companies centralise their clients in some sectors. Consequently, it might point out difficulties in responding to complex market conditions like the airline market. This idea supports the utility of the integration of multiple third-party fraud detection services to support different transactions and company needs. At the same time, it reinforces the idea that companies with different market segments might benefit from a system that allows the configuration of the risk assessment accordingly.

Table 2.3: Identification of market specialization for existing fraud detection service in the market

Fraud Detection Service Name	Market
Cybersource	Airline Industry and Others
Fraugster	Payments, Services, E-ccomerce
Ravelin	Services. E-ccomerce
SaltPay	No specialization
MaxMind	No Specialization
Radar	No Specialisation
Adyen	Service, E-ccomerce
Seon	E-ccomerce, Payments, Airline Industry
Fraudio	Payments

To conclude the study of the available services, we understood that the complexity and variety of results could be an issue for our implementation. For example, the existence of some outliers with a lack of global-scale assessment. Moreover, the support of customer input in the third-party fraud detection service will add complexity to implementing the integration solution. However, the services segmentation supports our argument, and these circumstances are ideal for developing a tool that supports customer configurability concerning conditions to run each fraud detection service.

2.3 Fraud Detection Combination

Our problem involves combining the execution of different fraud detection services in the same transaction; this way, we analyse different fields where the management of several elements is achieved. We focus on two subproblems: the execution and aggregation of results.

We start to explore orchestration solutions and focus on the execution process that our solution should support in Figure 2.4. Hence, we assessed workflow and rule engine solutions in search of solutions applicable to our problem. Workflow engines are solutions that control the execution of workflow-based sort of processes and are commonly applied in business processes. Rule engines execute the tasks when the conditions defined are met.

In [35] a clinical decision support solution with an architecture combining a workflow engine and a rule engine is presented. In terms of execution, the rule engine works as a point of invocation, and the correct rule is triggered based on the starting conditions of execution (the patient condition). After, the workflow decides the path of execution based on the outcomes of the rule execution phase (clinical decision). We can retain some points from this solution, but we focus our interest on the rule engine because, in our problem, we have conditions that influence execution paths. We should use their results to cyclically check the execution paths until no more routes are available to match.

Furthermore, looking at the existing solutions inside the SaltPay system, a rule engine system is already applied to choose the payment methods dynamically. The strategy applied is based on priorities. When executing, it will check the system status against the rules defined by the user ordered by priority (e.g., if the rule with priority one does not apply, it will check the rule with priority two). This priority system can be a strategy to tackle our problem. However, for this solution, the definition of the execution flow based on prioritised rules might be challenging to represent to the user visually. We should look for abstractions that are easy to perceive when asking for its input and focus on a validation process on its usability.

Advancing into the result aggregation, we have to consider a solution for the result aggregation of the outputs of the third-party services to reach a unique risk assessment classification. Since result aggregation is an explored concept in ensemble learning and logical systems, we will explore existing solutions in these study fields with the potential usefulness for our context.

2.3.1 Ensemble learning

Ensemble learning solutions' main goal is to combine several models, solving the same original task to obtain a better composite global model with more accurate and reliable estimates or decisions based on a single evaluation [45]. Similarly, we intend to combine several results of fraud detection models to achieve a global result that does not lose the value of the individual inputs. Ensemble learning has two categories based on discrete or continuous values: classification or regression, and as said in [24] classification can be regarded as a particular case of regression. In section 2.2.2 we show that fraud detection services provide both discrete and continuous results, but we can make all the results correspond to a single category with applied transformations.

Moreover, the application of ensemble learning techniques in risk assessment is already present in the literature, with many applications in the financial field regarding credit risk assessment. In [52], the authors apply a multistage neural network ensemble learning model to evaluate credit risk assessment. We focus on the final stage, where the ensemble theory combines the results. The proposed strategies for integrating the elements are maximum strategy, minimum strategy, median strategy, mean strategy, and product strategy.

Similarly, [50] proposes a credit risk assessment tool with a new hybrid ensemble approach, called RSB-SVM, which is based on two popular ensemble strategies and where the last approach applied to combine the results is a majority vote strategy.

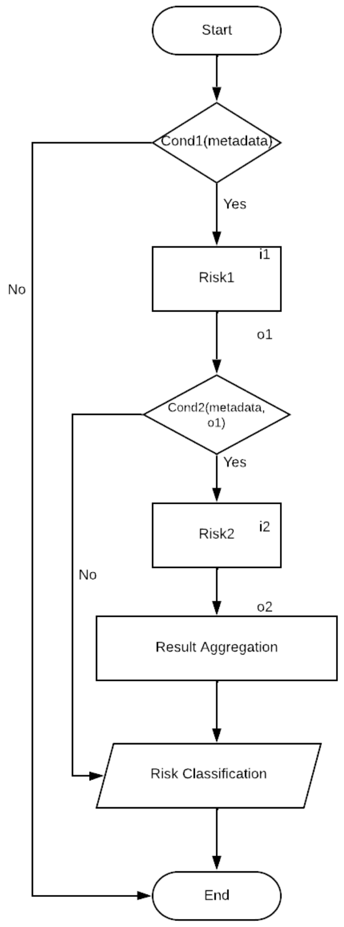


Figure 2.4: Flowchart for an example of serial orchestration for two third-party fraud detection services

Nonetheless, we do not intend to focus our study on creating a fraud detection score based on the transaction characteristics. In contrast, we want to reach a risk score based on various fraud scores from different external services. In fact, we want to remove this responsibility for the companies. This is why we focus our study on the last stage of ensemble learning, where the inputs are already several scores, and the goal is to combine them to create the most accurate result. Hence, this technique is interesting to apply in the risk assessment result aggregation.

2.3.2 Fuzzy rules

As fully elaborated in [27] fuzzy rules *“are rules whose antecedents, consequences or both are fuzzy rather than crisp.”*. Moreover, fuzzy rules are applied in problems of combining multi-classifiers into one result. Hence, we can adapt these solutions to our problem by considering the risk assessment results of the fraud detection third parties and the different classifiers.

Looking for concrete solutions, [48] points out that there are successful solutions in combining the individual classifiers in a multi-classification tool employing a fuzzy aggregation operator based, but there was a lack of combination method based on fuzzy rules. Accordingly, the article defines a successful combination method for fuzzy rule-based multi-classifiers. This solution uses fuzzy rules by attributing a certain degree to their consequences and applying a genetic algorithm to obtain the final result. Even though the complete presented solution might be challenging to accomplish given the time scope of implementation, the usage of fuzzy rules with a degree of consequence might be helpful to consider.

2.4 Data Warehousing

At present, significant operational databases are available in most companies. These databases may provide a significant wealth of helpful information. Decision support systems provide in-depth analysis of a company's business to accommodate faster and better decisions. This reality presents an excellent opportunity to develop a monitoring dashboard to work as decision support of the risk system and consequently support changes to its configuration. Thus, in this section, we explore topics of data warehousing that will back our work.

Business intelligence provides strategic decision support and transforms company data into actionable information with different detail levels. Strategic decision support is a sub-topic in decision support that includes evolution analysis and forecast, identification of critical business areas, budgeting, and identification of winning strategies where the cost is reduced and profit increased. Data warehousing is vital to *"support business intelligence (BI) activities, especially analytics"* [4].

A data warehouse is a type of data management system devoted to BI that performs queries and analysis for a specific subject and is kept from company operational databases. The data available in these systems is devoted to a specific subject, integrated and consistent, and time-dependent.

We can represent data with a hypercube or OLAP cube model with three or more dimensions. This model helps understand the logical model behind the data transformation. Following the example in Figure 2.5, we have a supermarket chain represented by three dimensions: shops, products, and date. Each cube (intersection) inside the model represents a product sale [31].

The process of designing a data warehouse is also fundamental to present in this document. It comprises several stages defined in [4]: requirement analysis, conceptual design, and logical design.

The requirement analysis collects data analysis requirements, application requirements - understand relevant facts for the business context and the workload necessities, e.g., periodicity of business reports - and structural requirements - understand available resources, architecture, and deployment planning. After, and supported by the defined user requirements, the next step should be the conceptual design stage, where the dimensional fact model is usually adopted.

The dimensional fact schema is a graphical model defined in [29] supporting conceptual design where it defines a fact schema modelling for a given fact. It models the information in facts

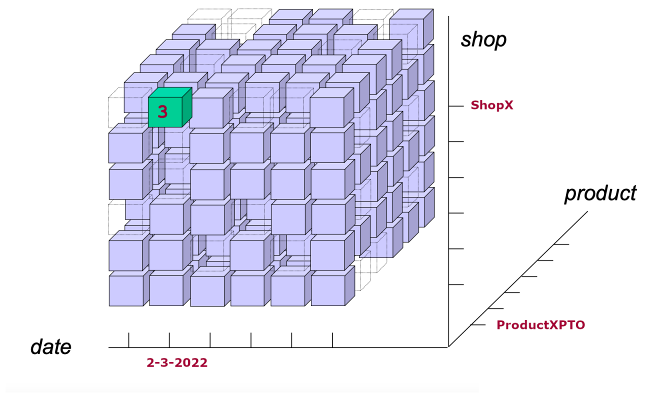


Figure 2.5: Hypercube for a supermarket chain data warehouse::adapted from [30]

with dimensions and measures. A fact models a set of relevant events and evolves with time. A dimension describes the different perspectives for analysing a fact and is typically characterised by categorical attributes. Measures describe a numerical property of a fact. In Figure 2.6, we have a representation of a fact schema example for a sales data warehousing system.

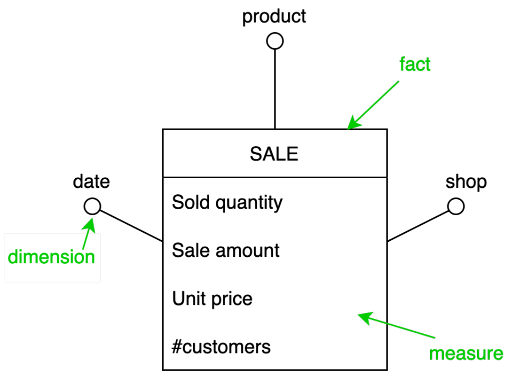


Figure 2.6: Example of a fact schema for a sale data warehouse model

The last phase is the logical design. With the input of the previous work done - conceptual fact schema, workload, data volume, system constraints - we define the relational, logical schema of the database.

2.5 Usability Evaluation

We should aim for a highly usable solution available to our users. Thus, we should focus on exploring the best ways to evaluate the usability of our interface. To do so, we will explore human-computer interaction concepts, more concretely the process of user evaluation or usability testing.

User evaluation has an extensive definition though the main goal is the problem identification inside our design to correct them as soon as possible. There are several approaches to the evaluation that might take place in the lab or the field. The evaluation in the lab provides an advantage when there is specialised equipment involved that is only available on-site; besides, it is an uninterrupted environment. However, it lacks context, making it harder to observe several users cooperating. It is valuable if the environment is unsafe or the location is impractical. In the case of the evaluation in the field, it provides the natural environment with the context available. In contrast, it can have distractions and noises, so it should be chosen when the context is crucial for the experience.

There are two experimental methods for user evaluation: usability testing and controlled experiences. The main difference is that the first is observation driven while the second is hypothesis-driven. We believe that a global evaluation of our design will benefit more from usability testing, so we will focus on exploring this topic.

"Usability testing speeds up many projects and produces cost savings in a system development" [46]. We should look for participants that represent the target user, and we should complete some background characteristics like the level of computing skills and experience with similar tasks; *"motivation, education, and ability with the natural language used in the interface, etc."* [46].

2.5.1 Testing stages

The usability testing has three stages: plan, run, and analyse the results. In the planning phase, we should focus on: choosing the participants, assigning roles, defining tasks, choosing the methodology and success criteria, selecting the equipment, writing a test protocol, and preparing necessary consent forms.

To choose the testers, we should define the target users of our system and complete the list of characteristics mentioned above for each user. Moreover, according to a highly regarded study from Nielsen and Landauer in [39] the number of participants for a qualitative usability test should be 5. It deconstructs the idea that more users will provide better results in qualitative testing using as the main argument the return on investment: *"testing costs increase with each additional study participant, yet the number of findings quickly reaches the point of diminishing returns. There is a little additional benefit to running more than five people through the same study; Return of investment (ROI) drops like a stone with a bigger N."* [37] Figure 2.7 and Figure 2.8 present the outcomes of [39] that support the premise above.

Another essential detail is each element's role: at least one facilitator and 1 to 2 observers that should also take notes.

The tasks to ask the participants to perform should be introduced with a scenario that should be objective and clear, and the number of tasks should go between 5 to 10 [32].

Possible methodologies are think-aloud, where the participant expresses what is thinking while performing the tasks asked; cooperative evaluation, where the participant and evaluator collaborate during the evaluation [22, 46].

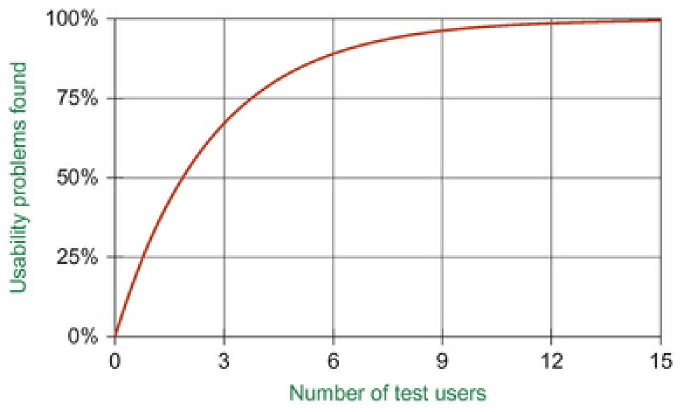


Figure 2.7: Evolution of usability problems found by number of evaluators in usability testing [22]

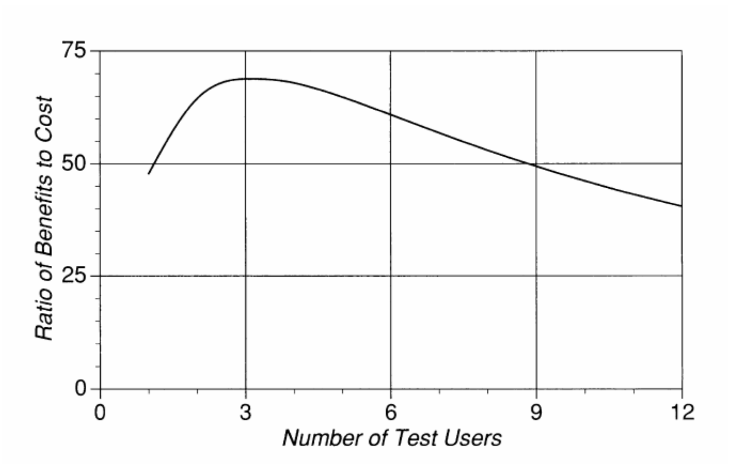


Figure 2.8: Evolution benefits to cost by number of evaluators in usability testing [38]

Subjective metrics and quantitative metrics can define the criteria for success. Depending on the type of assessment, the subjective metrics can comprise background data, user opinion, and satisfaction with the tested page. Those are associated with the end of the task or the testing exercise. Quantitative metrics are the number of successful completions, non-critical errors, time on each task, etc. For quantitative metrics, there are questionnaires for post-task, e.g., Single Ease Question (SEQ) or post-test evaluation, e.g., system usability scale (SUS) that are quantitative scales [34]. SUS measures the perceived usability of a system based on ten questions presented in Table B.1. It produces a score from 0 to 100. However, the only inference we can make from the result is that above 68 is considered above average, meaning that we cannot make a relation between the usability and a higher or lower value, given that this scale does not represent a percentage value [26].

In the test execution stage, all the prepared elements are put into practice. Furthermore, the facilitator should follow the script and remain neutral. It should also encourage participants to adopt the chosen methodologies. The observers should take notes on the participant behaviour,

Table 2.4: SUS Questions

1	I think that I would like to use this system frequently.
2	I found the system unnecessarily complex.
3	I thought the system was easy to use.
4	I think that I would need the support of a technical person to be able to use this system.
5	I found the various functions in this system were well integrated.
6	I thought there was too much inconsistency in this system.
7	I would imagine that most people would learn to use this system very quickly.
8	I found the system very cumbersome to use.
9	I felt very confident using the system.
10	I needed to learn a lot of things before I could get going with this system.

comments and the defined metrics for the success criteria.

After the usability tests, we should collect and summarise all the data available and look for trends across all the participants. The conclusions should help specify a plan for improving the interface.

2.6 Conclusions

In conclusion, several challenges and opportunities exist for developing a solution to integrate fraud detection services in payment processing systems.

Starting with studying the payment processes and leading players in the ecosystem, we comprehended crucial elements in the solution's definition, such as where in the payment flow is most appropriate to apply it.

Furthermore, with the fraud detection services study, we understood the variability of implementation for different market tools that will be important to define the correct approach to integrating third-party fraud detection services and maintain the most agnostic approach possible. At the same time, we were able to verify the versatility of the existing fraud detection services points of action. Since merchants have different groups of transactions, it supports our idea to allow merchants to configure the fraud detection services according to different transaction characteristics.

In searching for possible techniques for combining several third-parties executions with respective result aggregation, we went into research concepts that we could apply in our context, which provided solid insight into which a solution can be envisioned.

For the monitoring step, we focused on data warehousing concepts and had a better background in this field and possible methods to apply in our solution.

Finally, we focus on usability evaluation for our graphical user interface since it is our goal to create an intuitive interface for the communication between the system and the user.

The knowledge obtained from the research in this section contributed to the success of the final solution. We recognised many opportunities. To our knowledge, there is still no similar solution available, but many market solutions offer fraud detection services. Moreover, the solutions found in other field studies can be applied to our problem.

The main threat to the solution's success is the time available for its development, which might lead to some simplification decisions. Furthermore, the dependency on external services might make our validation process difficult, especially considering the privacy requirements to handle client transactions. Finally, the lack of similar solutions can be an opportunity and a weakness considering the lack of exploring the topic.

Thus, we considered the conclusions from the research discussed in this section when developing the final solution.

Chapter 3

Solution Requirements

This chapter will present the needs of our solution based on the research collected in the last chapters and the analysis of the current system and the needs of SaltPay. We start to go over the current SaltPay system, focusing on the high-level behaviour of the system flow to contextualise the solution inside the system. Afterwards, we explain the functionalities of the services that will interact with our solution and detail the existing risk assessment. Subsequently, we characterise the solution's requirements and detail some use cases to better capture the usage scenarios.

3.1 SaltPay Overview

The SaltPay payment gateway follows a microservice architecture comprising several interacting services that provide a unique solution to handle merchant payment transactions. This section presents how the SaltPay gateway deals with the transaction flow inside the platform and how the current risk assessment works.

To understand the transaction flow inside the SaltPay platform, we present an activity diagram in Figure 3.1. The first step occurs with the merchant's intent to realise payment. In this phase, the company sends the data collected (e.g., amount and payment processor) for validation to ensure that the configurations are valid with the merchant plan and create an element called charge containing relevant information for the next phase. Subsequently, the data is collected from the actual payment process, starting the next phase by running the risk assessment. We continue with the dynamic router that chooses the best payment processor to run the authorisation when the risk assessment results do not block the process. Based on the defined configurations for the merchant, the dynamic router might re-run the authorisation process with a new process in case of failure. At the end of this phase, an element called a payment instrument is created. The process finishes the flow with the merchant's actual capture of the customer's money and the payment creation.

Our focus inside the presented flow will be on the risk assessment stage. We will create a new tool to support third-party fraud detection services. Furthermore, the existing risk assessment system will support blocking the transactions tagged as fraudulent inside the payment gateway system.

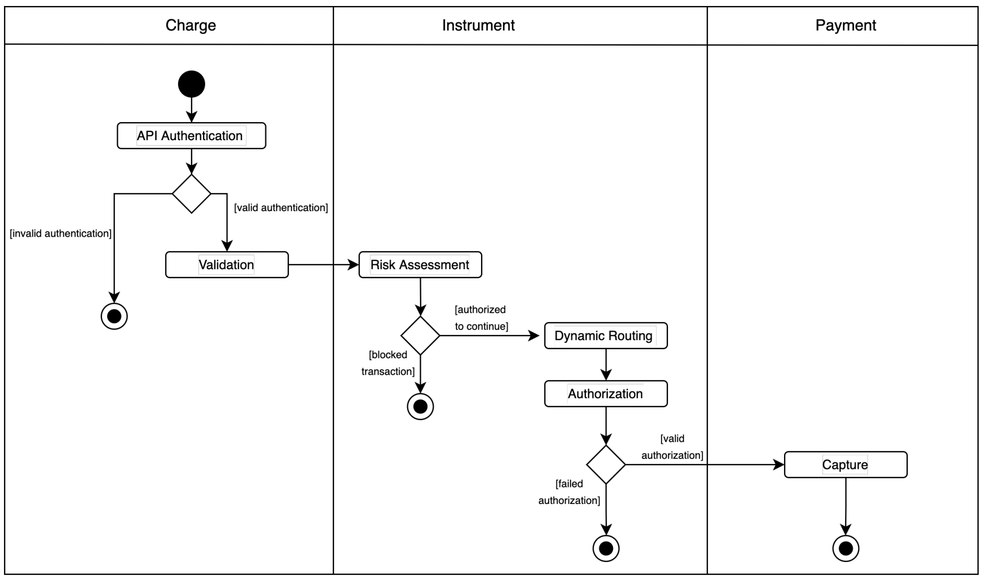


Figure 3.1: UML activity diagram of a payment flow inside the SaltPay gateway

3.1.1 SaltPay Services

As we already referred, the SaltPay payment gateway follows a microservice architecture comprising several interacting services responsible for the correct behaviour of the payment gateway. Thus, another essential aspect to proper grasp is to understand which services are relevant to our solution and how they interact with each other. This way, the services detailed are Switch, Risk, Merchant, Lifecycle modules, and Dashboard.

Switch module is responsible for the core flow of the payment gateway and the management of the payment processors. It is responsible for the payment processors' integration, and the main flow explained in Figure 3.1 is handled by this API calling the remaining services when appropriate e.g., to start the risk assessment, the Switch will interact with the Risk module to start that process.

In turn, the **Risk** module handles the transaction risk assessment, but we detail its behaviour in section 3.1.2.

The **Merchant** module is the API used to expose the internal endpoints for merchant usage safely. In other words, an endpoint exposed for external users will pass first through the Merchant API that will redirect the request to the proper service after authentication checks.

Another relevant component is the **Dashboard**. It allows customers to configure and visualise their operations inside the Salt platform. The current risk assessment is configured in this platform by providing an interface to create rules according to the outcome actions. Additionally, the merchant can see the action-outcome for each transaction that passes through the risk assessment.

Finally, **Lifecycle** handles events with the support of a message broker to ensure data consistency between all the services and databases.

3.1.2 SaltPay Internal Risk Solution

The SaltPay internal risk assessment is a rule-based mechanism. The main idea behind this mechanism is to allow the customer to flag transactions based on their characteristics. This way, the merchants select characteristics to create rules and associate them with a risk of fraud represented in the system by the actions: allow, block or review. However, this mechanism has a setback regarding the association of characteristics with direct risk outcomes since the merchant must have full knowledge of the fraud behaviour within its organisation to achieve satisfactory results.

We represent the internal behaviour of the Risk module in Figure 3.2. When the POS sends the payment information to our platform, the Switch module sends it in a request to the Risk module so that it can start the risk assessment. In this process, we start by collecting data that can be associated with the transaction that might be already in the system from past operations. Afterwards, we start matching the transaction information with the existing merchant rules from the database. As a result, we get an action from the risk assessment that we update in system metadata and send back to the Switch module, where the rest of the transaction flow continues.

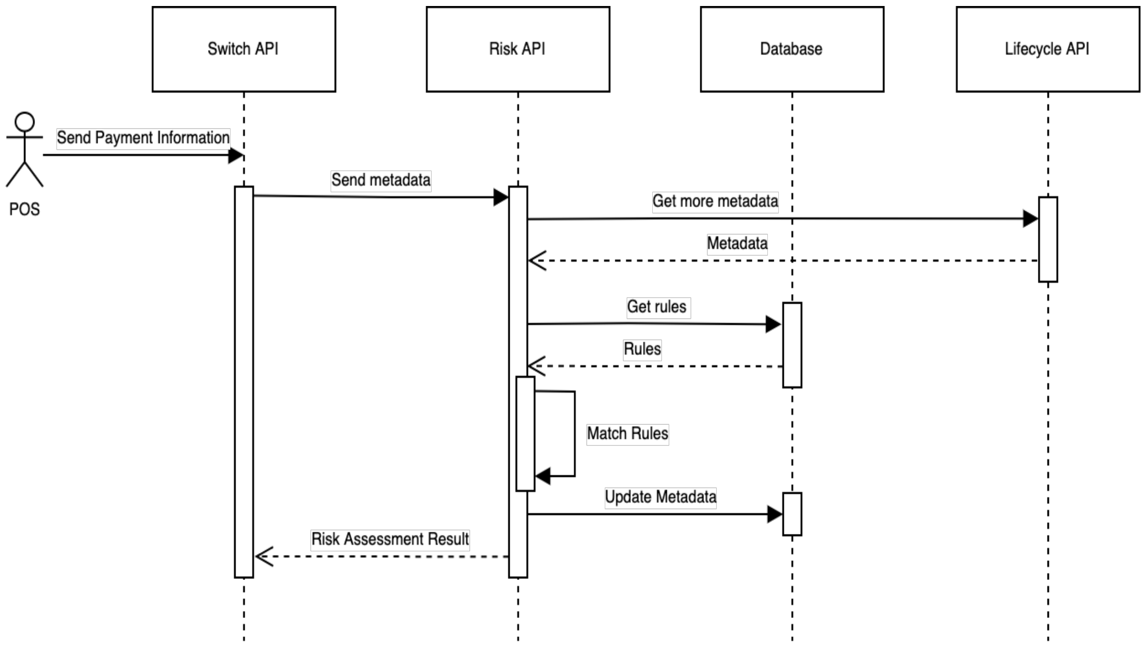


Figure 3.2: UML sequence diagram of the current risk service execution

3.2 Requirements Analysis

As already mentioned, the main goal of the new risk assessment system is to provide a complete solution to the customer that provides configuration, execution, and monitoring of the system. At the same time, it provides the merchant decoupling of responsibilities inside the fraud detection process since it passes most of this responsibility to third-party services. The merchant is only responsible for analysing the best fraud detection service for each group of transactions with the support of the monitoring data provided by the system.

This section formalises the aforementioned concept in smaller properties that the system needs to provide to accomplish it. We present the requirements with the definition of the features and system criteria below using functional and non-functional requirements, respectively. Additionally, we designed mock-ups to visually represent the requirements needed in the existing solution interface and are represented in the appendix A.

3.2.1 Functional Requirements

Starting with the functional requirements, we characterise all the system functionalities needed to accomplish the complete functional solution.

- **F1 - Integrate third-party fraud detection services**

  Integration of a couple of third-party fraud detection services. This functionality will be implemented in the Risk module and serves as a starting point to include the third-party calls in the transaction flow. In this step, the calls to the third parties will be made based on a queue independently of any rule system.

- **F2 - Combine several fraud detection services calls for a single transaction**

  Creates an orchestration system that calls the third-party fraud detection services based on a system of condition-action rules. The conditions are related to the transaction characteristics, and the resulting actions are a request to a third-party fraud detection service.

- **F3 - Aggregate the several fraud detection services results into a final result**

  Aggregates the results of the third parties in a single internal score that correctly represents all the values collected from the external fraud detection services.

- **F4 - Prepare a Risk Management Interface to let the customer configure the system process**

  Create a UI interface to support the customer creation of rules to configure the risk assessment system accordingly to the desired flow.

- **F5 - Prepare a dashboard inside the Risk Management Interface with monitoring information about the risk assessment system**

  Create a web-based UI to provide the merchant monitoring information about the risk assessment system. The page should allow data filtering according to relevant categories such as time interval, payment type, instrument country, etc. The dashboard should provide data relevant to customers' decision-making and accordingly promote system rules and execution changes.

- **F6 - Provide the system configuration and monitoring data via endpoints**

  Provide all the system functionalities via endpoints to allow power users to manage the process as it best suits their needs.

3.2.2 Non-functional Requirements

- **NF1 - Response Time**

  The solution should respect the stipulated time requirements for payment transaction processing. SaltPay currently has a 5s timeout for the Risk service. However, we acknowledge that this period is too long to provide a good user experience and stipulated a limit of 1s in developing this solution.

- **NF2 - Interoperability**

  The solution should share information with other SaltPay services to integrate into the transaction flow. Furthermore, it should share information with third-party fraud detection services to receive the fraud risk assessments essential to its correct functioning.

- **NF3 - Usability**

  The Risk Management User Interface should be intuitive and not present any significant usage challenges to the user. The target users are merchants and, most commonly, customer success agents that will implement the requests made by the merchants.

- **NF4 - Scalability**

  The solution should respond to the growth of the number of processed transactions simultaneously. With payment transactions constantly growing, we should expect a system capable of adapting to the growing needs. Nowadays, the strategies to respond to system increased loading are associated with increased physical resources. However, we should follow a system design that considers scalability best practices to avoid increasing resources because of a poorly designed system.

- **NF5 - Availability**

  Payment processing systems work with real-time execution flows and demand high availability since the incapability of a merchant to process a payment transaction would have critical consequences for the business. Thus, we need to maximise its availability and ensure that no bottleneck or critical error exists in the system that could compromise this property.

3.3 Use Cases Definition

To show the purpose and usefulness of our solution, we defined three groups of use cases based on different elements of the system. These use cases will mirror the functional requirements already described but in a more intuitive way.

3.3.1 Risk Management Interface Use Cases

Starting with Figure 3.3 we illustrate the functionalities available in the system for the user. The user that will interact with the system can be a merchant or, more commonly, a customer success agent that will configure and extract the data requested by the merchant. Thus, one interface component is responsible for configuring the risk system (**F2** and **F4**), meaning the creation, edition, and deletion of the rules available. In addition, it is also responsible for visualising the risk channels available in the merchant account (**F5**). With a risk channel, we refer to a third-party fraud detection service that we call within the tool. In the other component, the interface is dedicated to monitoring the risk assessment execution data and filtering the information to extract more precise conclusions for a set of transactions with similar characteristics.

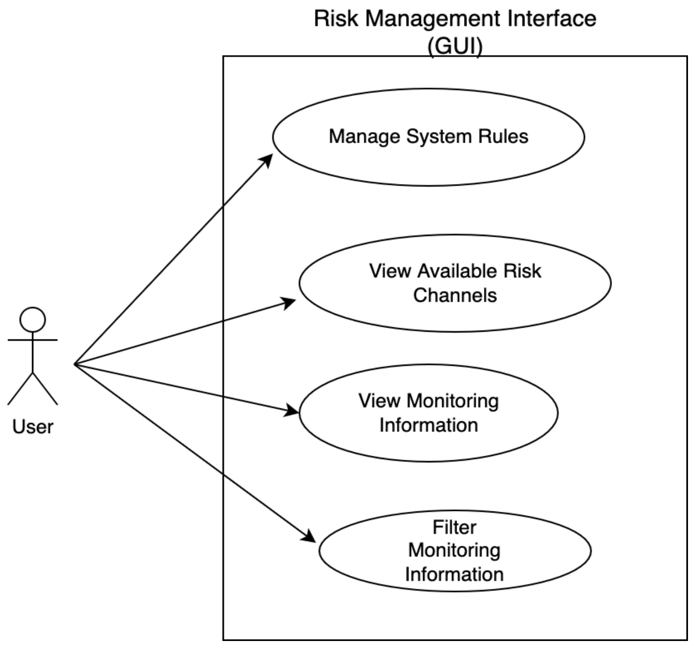

Risk Management Interface (GUI)
Manage System Rules
View Available Risk Channels
View Monitoring Information
Filter Monitoring Information
User

Figure 3.3: UML use case diagram for the Risk Management Interface

3.3.2 Risk API Use Cases

Going to Figure 3.4 we focus on the possible interactions of the user with the system via the endpoints available by the Risk API. The creation of a risk channel is exclusive to a super user referent to a staff member since it implies a contract between the merchant and third-party fraud detection service that the payment gateway staff should confirm before adding the channel inside the platform (**F1**). Besides, we have represented other functionalities in the Risk Management Interface. However, the direct endpoint availability to the user can be helpful for scripting purposes, e.g., continuous monitoring of a critical analytic value to automatically delete a rule calling a risk channel causing the value change (**F6**).

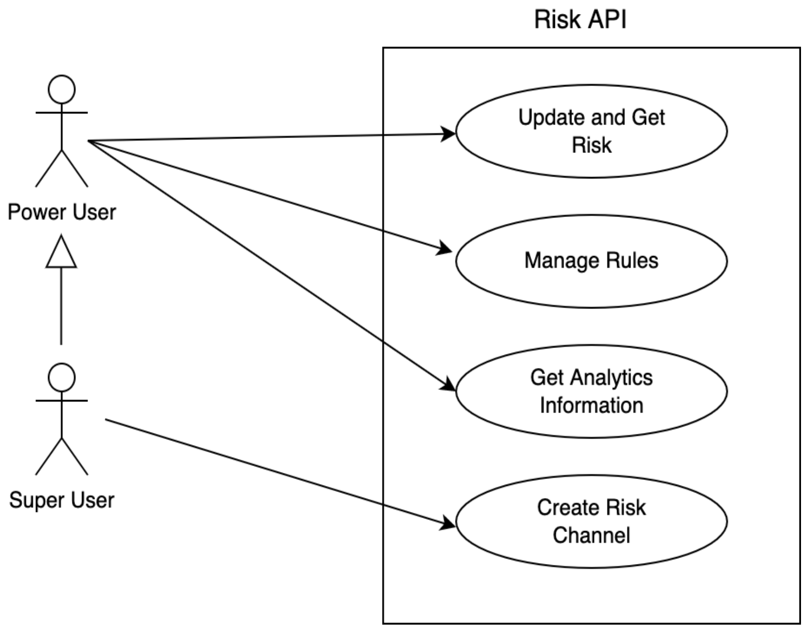


Figure 3.4: UML use case diagram for the Risk API

3.3.3 Third-Party Execution Use Cases

The last use case in Figure 3.5 refers to the interactions between the third-party fraud detection services and the Third-Party Execution component. The Switch API is responsible for the main flow of transactions. At the right moment, the Switch API will trigger a call to the Risk module to start the risk assessment and, consequently, trigger the Third-Party Execution System requests for fraud detection services (**F1**) that will later be aggregated into a unique *global risk score* (**F3**). Thus, we highlight that the functionality of the fraud detection services is dependent on the call of the third-party fraud detection services.

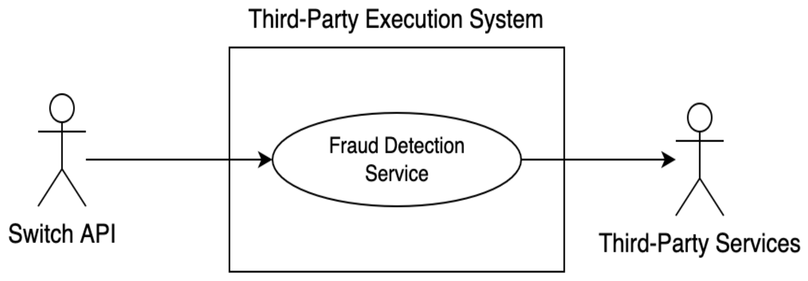


Figure 3.5: UML use case for the Third-Party Execution System

Chapter 4

Solution Conception and Implementation

In this chapter, we will go through the conception and implementation of the solution. Starting with the solution architecture, we detail in a component diagram the changes and new component additions to the current system-subsequently, we provide the workflow of the risk assessment with our added solution. Afterwards, we go into more detail about each new system component where we associate the functional requirements present in section 3.2.1 meet with each new functionality.

4.1 Solution Architecture

Our solution consists of a fraud risk assessment that enables customers to configure, execute and monitor the process. The system conditions are based on the transactions' characteristics so that the merchant can choose the best fraud detection service for each group of transactions in the business.

Presenting the solution components in Figure 4.1 we distinguish the existing services and components from those that were changed or created to develop the solution using special icons.

Following the transaction flow order, we start with the Switch service that remains unchanged; it triggers the Risk service to start running the risk assessment.

In contrast, the Risk is the service with more changes and new components since this is the component responsible for the risk assessment process. We created two new components: the Third-Party Integration to support the integration of individual fraud detection services and the Third-Party Execution responsible for system management of the third-parties requests that is the centre of the solution execution phase. Additionally, we had to incorporate the new components into the existing service's transaction flow, and we created new models inside the database to accommodate the new elements in the Risk service.

Passing to the Merchant API, this service makes the created endpoints available to the customers by ensuring that the requests meet the specified authentication requirements. At the same time, we make the data available to the new Risk Management Interface, which will be detailed below.

We created a user interface (UI) called Risk Management Interface for monitoring and configuration. It facilitates the user interaction with the system, abstracting the Risk endpoints exposed by the Merchant API by providing an intuitive graphical user interface (GUI). The interface is divided into two components: the configuration of risk rules and the monitoring dashboard that provides relevant analytics about the execution of the risk system.

The technologies used for the implementation are Python with Django for the backend and MongoDB for the database. We developed the frontend interface using ReactJS, Material UI, and an open-source library called Material Dashboard 2 React [13].

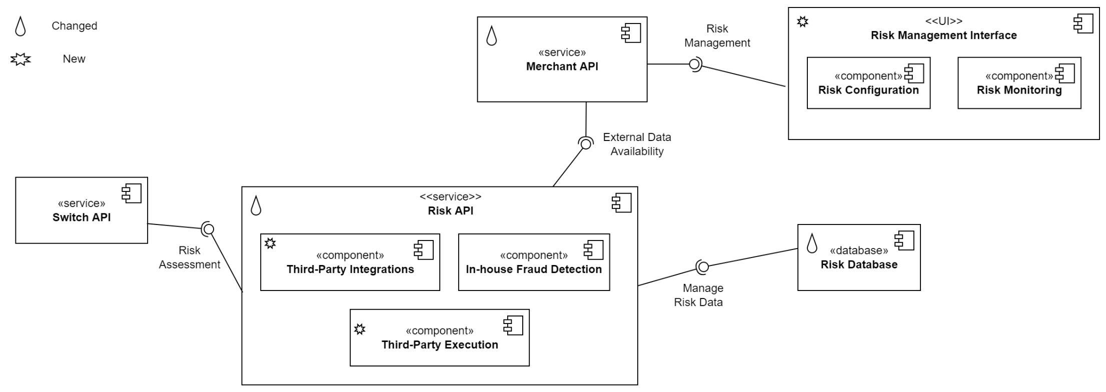

Changed
New
«service» Merchant API
Risk Management
<<UI>> Risk Management Interface
«component» Risk Configuration
«component» Risk Monitoring
External Data Availability
«service» Switch API
Risk Assessment
<<service>> Risk API
«component» Third-Party Integrations
«component» In-house Fraud Detection
«component» Third-Party Execution
Manage Risk Data
«database» Risk Database

Figure 4.1: UML component diagram for the implementation view of the proposed architecture

4.2 Solution Workflow

In Figure 4.2 we represent the new risk assessment workflow with our solution. When a transaction comes to the Risk service, it maintains the step of processing the metadata as described in section 3.1.2.

After this phase, we add a new stage to call the Third-Party Execution when active for the merchant. This option was made optional so merchants using the old solution would not have extra latency for a step they would not use. However, the activation of this stage is mandatory for our work. This way, going through the step, we verify if the current transaction metadata matches any rule. In a positive case, it will choose the rule with higher priority and call the associated fraud detection service. After, it goes back to match lower priority rules with the transaction and will be in this process until no more matches are found.

When we finish this step, the collected scores go through an aggregation process. Here, in case more than one fraud service is called, the system will aggregate all the scores to create the *global risk score* by applying the correct method for the concrete situation; e.g., if there are two dependent fraud service scores, we might consider only the last one. We explain the applied aggregation methods in section 4.5.1. As an output of this process, the system has a unique *global risk score* for the transaction, which will join the rest of the transaction metadata for the internal assessment.

Going into the in-house risk assessment, where we have a rule-based engine, we will associate the *global risk score* with an action that corresponds to the already defined internal logic of the SaltPay system. Using the existing system was not mandatory for implementing the solution in other environments, and similar logical approaches could be used. Nonetheless, it served as a simple and effective way to integrate the system result into the current risk assessment inside SaltPay. For the solution developed in this work, the old risk assessment should exclusively be used to support the described translation of the *global risk score* to an action in the system.

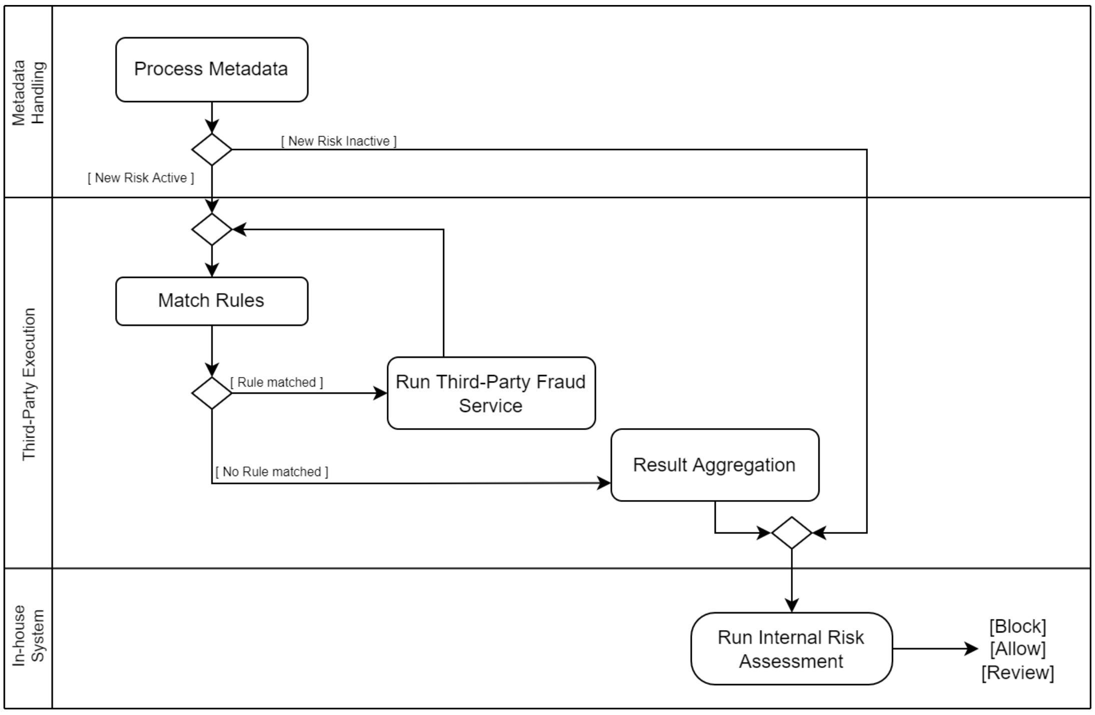


Figure 4.2: UML activity diagram for the sequence of actions inside the Risk module

4.3 Services Integration

To start our solution, we need to integrate the fraud detection services in the platform satisfying **F1**. Regarding business requirements, each merchant should only have access to the third parties with a contract defined. Because of this, the creation of new integrations by a merchant should be the responsibility of the SaltPay staff members. This way, it can ascertain that the merchant has the authorisation to use the fraud detection service. Additionally, in terms of business process, this restriction allows customer support to understand the client's needs and assist in the integration process to go as smoothly as possible. The process involves many technical details like sharing needed authentication metadata to communicate with the fraud detection services.

To define the integrations of fraud detection, we created a model in the risk database. Each model is the association between a fraud detection service and a merchant. It has information about the service configurations and is named within the platform as a risk channel. With this logic, to call each service, we consult the data available for the fraud detection service call that provide the appropriate metadata to complete the call to the third-party service, e.g., API keys. To handle the behaviour of each service call, the services implement an interface to respect a standard integration structure.

The creation of the functionalities above was made available via the following endpoints and helped fulfil **F6**:

- **POST merchants/{merchant_id}/channels** : risk channel creation; only staff members can use this endpoint and the restriction is applied based on the account authentication associated to each request
- **PUT merchants/{merchant_id}/channels/{channel_id}** : update a risk channel information; useful when the fraud detection service updates the data needed for the request
- **GET merchants/{merchant_id}/channels** : return all the channels available for a specific merchant
- **GET merchants/{merchant_id}/channels/{channel_id}** : return a specific risk channel available for a specific merchant
- **DELETE merchants/{merchant_id}/channels/{channel_id}** : delete a specific risk channel available for a specific merchant; currently, the historical data related to the channel is maintained in the transactions data. Nevertheless, as future improvement, we should create a delete flag in the data model to consider the channel as deleted but maintain the date of deletion available based on the last update;

4.4 Rule System

The implementation of the third-party system execution relies on a model of condition-action rules, meaning that for each risk assessment call, there is a condition that triggers its execution. To create this execution mechanism, we defined a rule model that will be used to support the orchestration flow. These rules will help satisfy **F2**.

4.4.1 Rule Definition Model

The condition-action mechanism applied follows the logical rule model:

if CONDITION then ACTION

The condition has the schema presented in Figure 4.3 where we can see the difference between the high-level logical model and the concrete syntax applied. The fields and operators defined are available in appendix B. Moreover, each rule has a unique priority associated, which will determine the execution order when more than one rule matches the current transaction characteristics. A smaller number corresponds to a higher priority in the system.

Following the rule schema we define a example rule:

1. if amount >= 50 then third-party-1

The condition is divided into three parcels: *amount* is the field, >= (greater than) is the operator, and the value is 50; the *third-party-1* represents the name of the third-party fraud detection service that would be triggered with this condition; *1* is the priority of the rule and has the higher priority in the system.

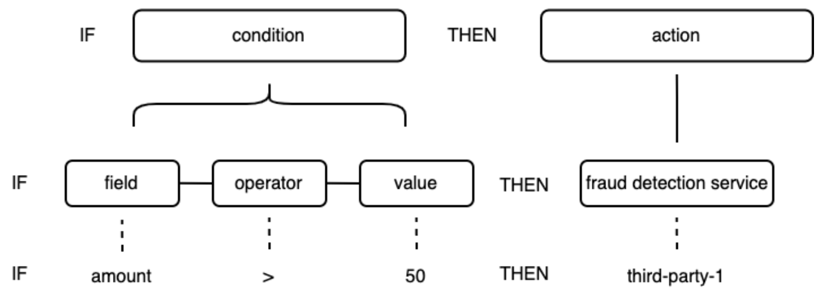


Figure 4.3: Rule logical model

An essential factor to point out in the rule system is the possibility of creating rules dependent on the result of previous fraud detection calls. These rules can be used to create serial execution for more than one service, like the example in Figure 2.4. Considering the different costs of fraud detection services, one might use a lower-cost service to filter the number of transactions that pass to the higher cost but higher precision service. This approach can be relevant because these services' costs are often associated with the number of transactions processed. Additionally, a company might offer the internal risk assessment inside this system with minimal costs compared to other services.

Another relevant feature is the creation of unconditional rules. This means that all the transactions that pass through the risk system will run the fraud service associated with this rule. This can be suitable for merchants who simply want a unique fraud detection setup without any transaction restrictions. The rule syntax follows the example below, and internally, the system is associated with the condition *"true"* in its field.

if any condition then third-party-1

To sum up, the goal of the rule system is to provide merchants with a way to group transactions based on characteristics that can be useful to support the decision of which fraud detection service to call. Hence, we defined the chosen fields considering the main factors that might influence this decision. We should consider that merchants are often interested in creating a lucrative system, meaning that they want to minimise the global costs of fraud detection. Thus, the rule-based execution system allows the merchant to do just that. The available transaction characteristics often mirror factors such as market differentiation, risk probability, the relevance of the value lost, etc. This way, we can list the rule condition fields by categories related to the transaction characteristics:

- Amount
- Location
- Frequency of usage
- Card characteristics
- Transaction characteristics
- Score of previous fraud detection services called
- Unconditional execution

However, we should always consider that the system's primary goal is to provide a configurable third-party-based risk system to the merchant that passes the risk assessment responsibility to the external fraud detection services. Thus, the configurations should focus on grouping transactions into relevant groups to call a given fraud service instead of serving as a risk system.

4.4.2 Rule Execution Model

We thought of several different approaches for the design of our rule model. In this section, we justify the reason for our choice by explaining the characteristics of our solution and how they compare to other approaches.

The rule's priority is relevant to our model's definition and simplifies the rule's conditions. This approach forces the person creating the rules to have attention to possible scenarios where a rule is never satisfied. For example, considering the system rules:

1. amount > 20 then third-party-1 (R1)
2. amount > 50 then third-party-2 (R2)

In this example, the second rule is never called. However, if we defined the model in the reverse order, this situation would not occur:

1. amount > 50 then third-party-2 (R2)
2. amount > 20 then third-party-1 (R1)

Assuming R1 and R2 as identifiers of the conditions that correspond to the group of values that respects their condition as true, the first scenario happens because the group of values associated with the condition on the second rule is a sub-set ($\subseteq$) of the group of values associated to the condition on the first rule $-R2 \subseteq R1$ - which means that for all the transaction characteristics x, the conditions R2(x) where R2 is valid, the conditions R1(x) for R1 will also be valid : $\forall x, R1(x) => R2(x)$. Since the priority enforces R1 to be chosen over R2, in every situation where R2 is valid, R1 will be chosen over R2. For the same reason, in the second scenario, the reversal of the priority order will solve the issue. When both rules match, the priority will enforce the choice of R2. However, R1 will still be selected in the cases where it does not match R2. An alternative approach would be only to allow exclusive rule's conditions like the following scenario:

- amount > 20 and amount < 50 then third-party-1 (R1)
- amount > 50 then third-party-2 (R2)

This approach would allow the removal of a priority since there is never a case where R1 and R2 will always be different sets: R1 $\not\subset R2 \wedge R2 \not\subset R1$. However, the rule's complexity would increase, and readability would decrease considerably.

Thus, we decided to opt for a system that allows non-exclusive rules but allows us to define priorities to handle possible inconsistent cases otherwise. Regardless, we understand that this model allows inconsistencies if we create the rules without previous thought. Nevertheless, given that the user does not update the rules with significant frequency and, when updated, are usually done with the supervision or support of someone with knowledge of the company system, we believe that simplifying the rule model was a better option to model the rules. In future work, we could define a more complex validation system to better block these inconsistent rules on creation.

With the rule execution model defined we also want to detail an example of serial execution based on the scores of previous fraud detection services. Thus, considering the example:

1. amount > 20 then third-party-1
2. third-party-1 score > 0.4 then third-party-2

For a transaction with an amount higher than 20, it calls the third-party-1. Consequently, the output score is normalised to an internal scale. Afterwards, when this score is higher than 0.4, it calls the third-party-2. Finally, because of the dependency between calls, we only consider the last fraud detection service call for the final *global risk score*. The aggregation techniques are better described in section 4.5.1.

Similarly to the integration section, the new rules functionalities are available in the API via new endpoints and help fulfil **F6**:

- **POST router/{merchant_id}**: create a router rule for a specific merchant

- **POST router/{merchant_id}/rules** : create router rules for a specific merchant; used for a bulk edit approach inside the Risk Management Interface
- **GET router/{merchant_id}** : get all the rules assigned to a specific merchant
- **GET router/{merchant_id}/{rule_id}**: get a specific rule for a specific merchant
- **DELETE router/{merchant_id}/{rule_id}**: delete a specific rule for a specific merchant

An important factor considered when developing these endpoints was the correct validation of the endpoints data. To consider a rule valid on creation, the following verifications are made:

- Check if the introduced field and operator respect the list of existing elements
- Check if the value type is appropriate to the chosen field
- Check if risk channel is available for the merchant
- Check if the priority of the new rule is valid based on the existing set
- Check if the rule condition is equal to one already present in the existing set

4.5 Execution Engine

The execution of the system is the piece that will combine the different components into a third-party functional solution and allow the fulfilment of **F2** and **F3** and corresponds to the component Third-Party Execution.

The system execution is an iterative process where we execute actions based on the matched rule in the iteration. The matching process checks existing rules that respect the current transaction characteristics and chooses the highest priority rule while discarding the others. Moreover, for each iteration matching process, we discard the already applied rules in the previous iteration execution. We resume this process in the following action points:

1. Match rules with current state
2. Choose higher priority rule
3. Execute action associated with the rule
4. Discard executed rule from the stack

After providing a high-level explanation of the execution model, we describe in more detail the implementation. As already mentioned in the solution architecture, inside the risk service, the solution developed will be called if it is active for the merchant responsible for the transaction. The execution mechanism manages the calls for third-party fraud detection services. It is responsible for overseeing the correct execution of the third parties based on the defined rules by the merchant explained in the section above. The logic behind the developed execution algorithm is presented in Listing 4.1.

```python
def evaluate_transaction(self, merchant_id, metadata):

    used_channels = []

    # Match rules with the metadata and return the biggest priority rule
    rule = RouterRule().match_metadata(...)

    while rule:
        # Get the information needed for the fraud detection endpoint call
        provider_name = rule['provider']
        channel_metadata = Channel().get_channel_by_provider(...).get('metadata')

        # Get the interface for the third-party fraud detection service
        interface = self.get_provider_interface(channel_metadata,...)

        # Call the third-party and return the risk assessment result from the fraud
             service
        risk_result = interface.evaluate_transaction(params=metadata,...)
        result = {'provider': provider_name,'risk_result': risk_result}

        # Update metadata with the third-party result
        metadata['providers'].append(result)

        used_channels.append(provider_name)

        rule = RouterRule().match_metadata(used_channels,...)

    # Aggregate the risk assessment results from the fraud services into a single
        result
    global_score = Processor.result_aggregation(...)

    return metadata, global_score
```

Listing 4.1: Execution algorithm of the orchestration solution

The merchant rules' existence shapes the execution algorithm since the call of each provider is defined in them. For each loop executed, the algorithm will check *a priori* if there is any rule that matches the transaction metadata and selects the rule with the most significant priority. This rule will dictate the third-party fraud detection service to call. We append each result to the transaction metadata since the defined rule's conditions might depend on third-party call results. Furthermore, the function used to select the following rule to execute needs to consider the already called third-party fraud detection service so that it will not choose the same more than once in the same run.

When no more rules match the transaction, we exit the loop and apply an aggregation algorithm that will return an internal global result based on the third-party fraud detection service.

4.5.1 Result Aggregation

After managing the execution of several third-party fraud detection services, we had to create a mechanism to aggregate the external results into a single value that responds to **F3**. We made this decision since we want to abstract the complexity of combining several fraud detection services outputs while still providing the meaning that their results provide.

The global score will be a continuous score from 0 to 1, representing a probabilistic value for the risk of fraud in a transaction and is defined as *global risk score*. A higher value represents a higher probability of the transaction depicting a fraud.

Furthermore, we defined a default risk level scale where we divided the *global risk score* into three levels of risk of fraud as represented in the following Table 4.1. This scale provides a reference for the actions for each probability score. Ultimately, in future improvements, the customer should still have the flexibility to decide which *global risk score* values correspond to which actions.

Table 4.1: Division of the *global risk score* values into three risk levels for the transaction

Global Risk Score	Risk Level
0-0.4	Low Risk
0.4-0.7	Medium Risk
0.7-1	High Risk

For the cases of serial execution where one fraud service is dependent on the result of another, we decided to consider the result of the last third-party called. In contrast, we calculate the mean value for all the probabilistic results for serial execution of independent calls, hence giving all the scores the same weight for the final result. For future improvements, it could be interesting to allow the merchant to choose the aggregation methods for each situation, e.g., default, maximum or minimum score.

In Listing 4.2 we show the aggregation algorithm inside our tool. We start by searching for the rules that depend on other risk channels. After, we extract the risk channels in which they are dependent on removing their score from the final result, as explained above. For the remaining scores, we apply the formula below for the *global risk score*:

$$globalriskscore = \sum_{i=0}^{n} \frac{score(i)}{n}$$

```
def result_aggregation(rules, score_results):

    if not rules:
        return -1

    aggregate = 0
```

```
        dependent_channels_to_remove = []

        for rule in rules:
            rule_dependencies = RouterRule().check_rule_dependencies(rule)
            if rule_dependencies is not None:
                dependent_channels_to_remove.append(rule_dependencies)

        to_aggregate = list(filter(lambda x: x['provider'] not in dependent_channels_to_remove, score_results))

        if not to_aggregate:
            return -1
        n = len(to_aggregate)

        for s in to_aggregate:
            if s.get('score'):
                aggregate += 1 / n * s.get('score')

        return aggregate
```

Listing 4.2: Algorithm for the result aggregation

To create the global score, we start to normalise the outputs provided by each third-party fraud service into a value in the applied scale for the *global risk score*. Because of the disparity and variability between the outputs of different third parties, we had to complete this normalisation process at each integration level. We can divide the normalisation process into two categories based on the data types of the results:

- **Continuous outputs**: when necessary, a correspondence between the output scale and the internal scale will be applied with a mathematical normalization formula:
  $normalized\_score = \frac{X - X_{min}}{X_{max} - X_{min}}$
- **Qualitative outputs**: based on the number of levels in the scale, we apply a correspondence to a percentage level in the internal scale. For example, using the Ravelin scale detailed in section 2.2.2 we could apply the following conversion: we associate each defined category to an internal risk level: allow to low, review to medium and prevent to high. The *global risk score* value to assign is calculated based on the median of each risk level scale. This way, allow would be converted to a 0.2 score, review to a 0.55 score and prevent to a 0.85 score.

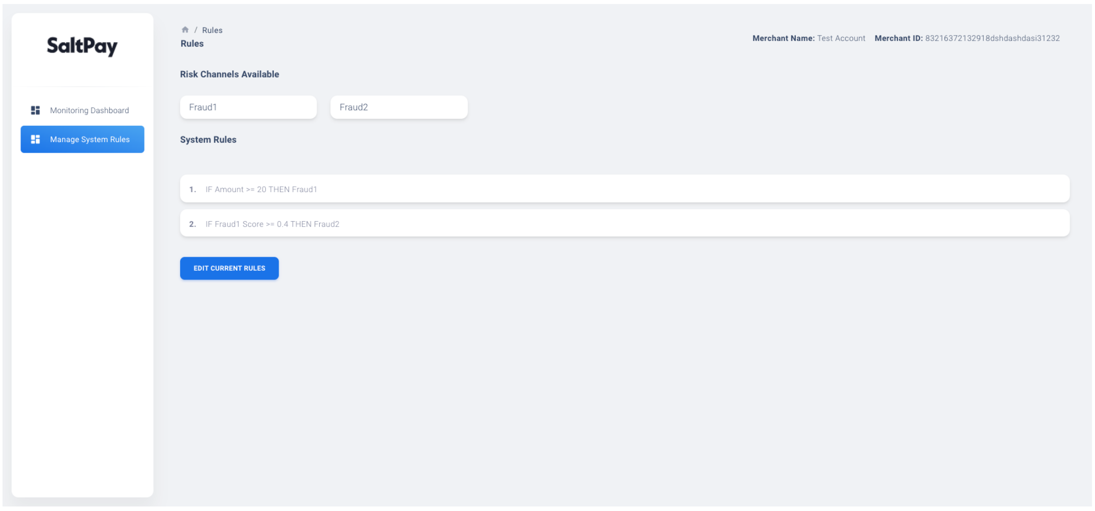
4.6 Risk Management Interface

Recapping, we already explained the system rule model and how it conditions the system execution. After that, we detailed the execution algorithm and how we aggregated the results from the different fraud detection services.

This section will discuss the Risk Management Interface available for facilitating user interaction with the elements already defined. Furthermore, we detail the monitoring functionalities inside the interface that are still to be discussed in this document. We divide the platform into two main processes: managing the rule system for the execution of the third parties and monitoring the execution results.

4.6.1 Risk Configuration

The risk configuration provides a GUI to manage the system rules with information about the available risk channels to facilitate the comprehension of the current state of the merchant account. The main goal is to provide a more intuitive way for the user to communicate with the system instead of direct interaction with the created endpoints. It respects **F4**.



Merchant Name: Test Account **Merchant ID:** 83216372132918dshdashdasi31232

Risk Channels Available

Fraud1 Fraud2

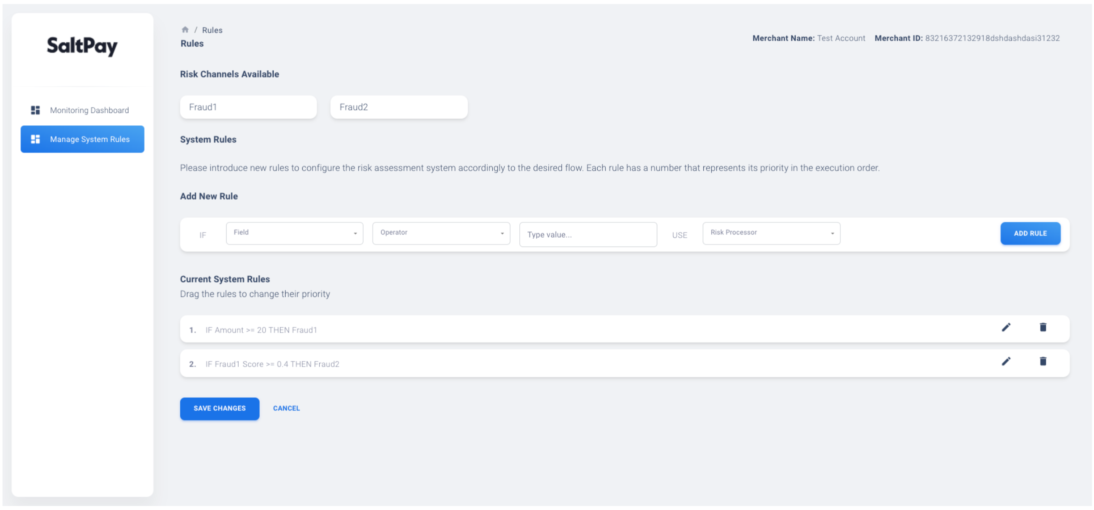
System Rules

1. IF Amount >= 20 THEN Fraud1
2. IF Fraud1 Score >= 0.4 THEN Fraud2

Figure 4.4: Risk Configuration Page in the Risk Management Platform

This page has a default view shown in Figure 4.4 and an edit view. To start making changes to the rules, we should click on the button with the description "Edit Current Rules," which will take us to the edit view available in Figure 4.5.

We can create new rules and edit or remove existing ones. All the changes will be communicated to the backend when the user saves the changes in the "Save Changes" button. This bulk edit approach allowed the user to experiment with new rule configurations before submitting the changes. We made this choice since the rules should be defined logically together, given the relationship they can have with one another. Hence, a POST endpoint with all the new rules is used when the submission happens. Besides, this approach allows the user to easily cancel all the changes made and return to the last configuration, respecting one critical usability criteria.



SaltPay

Monitoring Dashboard

Manage System Rules

Rules

Rules

Merchant Name: Test Account **Merchant ID:** 83216372132918dshdashdasi31232

Risk Channels Available

Fraud1 Fraud2

System Rules

Please introduce new rules to configure the risk assessment system accordingly to the desired flow. Each rule has a number that represents its priority in the execution order.

Add New Rule

IF Field Operator Type value... USE Risk Processor ADD RULE

Current System Rules

Drag the rules to change their priority

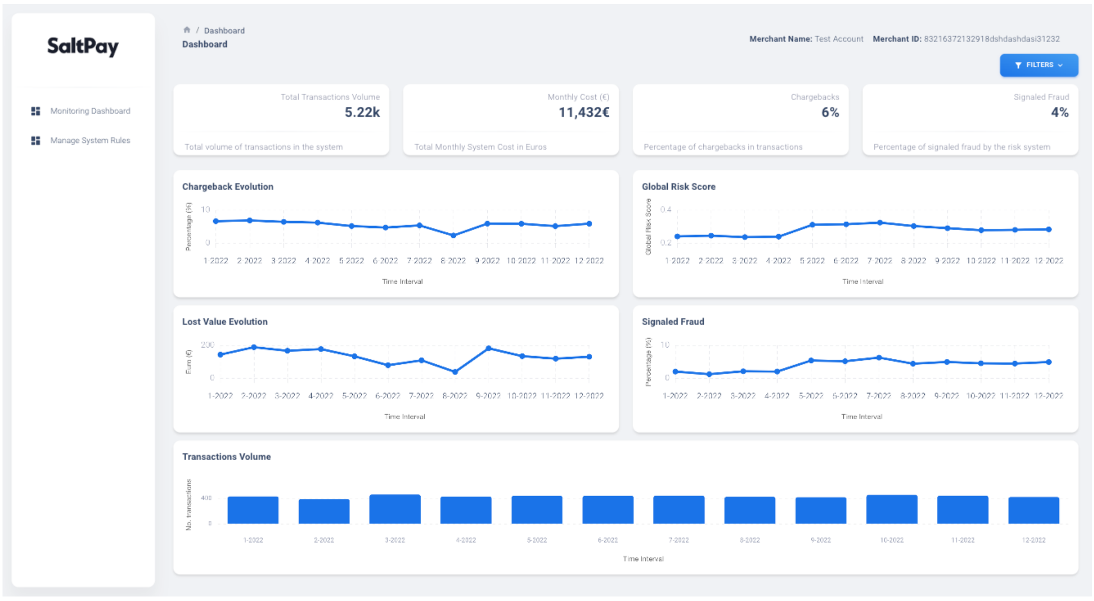
1. IF Amount >= 20 THEN Fraud1
2. IF Fraud1 Score >= 0.4 THEN Fraud2

SAVE CHANGES CANCEL

Figure 4.5: Risk Configuration Page in the Risk Management Platform - Edit View

4.6.2 Result Monitoring

Another critical aspect of risk management is the data monitoring of the system execution. The provided analytics create a decision support system. Hence, the customer can have an output over the quality of the business operations and use this knowledge to improve it if needed. The monitoring board presented in Figure 4.6 shows the page in question, fulfilling **F5**.



SaltPay

Monitoring Dashboard

Manage System Rules

Dashboard

Dashboard

Merchant Name: Test Account **Merchant ID:** 83216372132918dshdashdasi31232

FILTERS

Total Transactions Volume: 5.22k — Total volume of transactions in the system

Monthly Cost (€): 11,432€ — Total Monthly System Cost in Euros

Chargebacks: 6% — Percentage of chargebacks in transactions

Signaled Fraud: 4% — Percentage of signaled fraud by the risk system

Figure 4.6: Risk Monitoring Dashboard

4.6.2.1 Logical model and technologies

To develop the analytic functionalities, we used data warehousing domain knowledge since it matches the needs of our project. We wanted to create a base of historical data that can be retrieved and analysed to provide helpful insight into the third-party execution operations. With this in mind, we had to study which transaction characteristics were most important to this evaluation, summarised in the fact schema proposed by [29] defined in Figure 4.7. In our context, the merchant dimension is always applied, and the interface allows to filter the data by the following dimensions: acceptor country, instrument country, date, payment type and provider group name. We can retain information about all the measures available in the transaction for the different detail levels. Furthermore, the dashboard elements associated with chargeback and signalled fraud filter the data by another detail level - the transaction outcome.

This way, we can operate the data available at different detail levels based on the represented dimensions. We can apply the changes to the detail levels with the different filters for the data. For example, we can group the available transactions based on the "chargeback" outcome and extract the average global risk score, amount and lost amount, which are measures of the transactions.

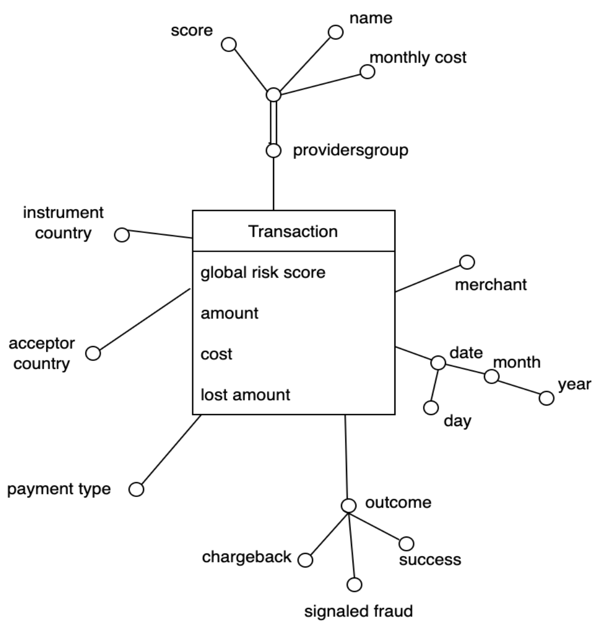


Figure 4.7: Fact Schema for the Data Warehouse

In terms of implementation, we opted to use the database available for the Risk service in MongoDB since this database model is appropriate for analytic data extraction and the data required was already defined there and correctly handled inside the flow of the risk service. MongoDB is appropriate for small to medium data storage and processing, which would not meet the solution requirements' long-term necessities. Thus, we would advise considering more scalable solutions when replicating this system. However, given the project time constraints and the functionalities of MongoDB, we considered it a good option to prototype our idea.

For example, Apache Druid is an open-source database with a profile that matches the referred needs and is a *"power real-time analytic workloads for event-driven data, and is not a traditional data warehouse."* [5]. At the same time, traditional data warehouse solutions are also adequate in this scenario. As presented in section 2.4, a data warehouse implementation would imply the isolation from the original data sources to focus the performance on the responsible analytic workloads. Ultimately, the decision should ponder the internal business requirements for developing the system.

4.6.2.2 Data

After explaining the decisions behind the chosen technological approaches to the problem, we will detail which queries we defined to extract the needed insights from the database and our monitoring dashboard. To do that, we focus on the data made available in the dashboard, referencing Figure 4.6 again. Observing the dashboard, we divided the data by accumulated values and time-series graphs. The time series are represented with a monthly granularity; it could evolve to a configurable parameter in the future.

The evaluated values are:

- **transactions volume**;
- **chargebacks**: a transaction where the payment is reversed after a customer disputes a charge on their account statement and is usually associated with poor services like non-delivered items or fraudulent transactions. We consider only the chargebacks associated with fraudulent activity in the dashboard. Nonetheless, SaltPay should implement this distinction in future work by analysing and categorising the existing chargebacks based on the chargeback reason;
- **signaled fraud**: blocked transactions by the risk assessment service;
- ***global risk score***: described in 4.5.1;
- **lost value**: value lost in each transaction when chargebacks occur;
- **system cost** - the cost of the system, which varies with the third-party configurations chosen;

Similar to the other project functionalities, the data is available in the following endpoints and fulfil **F6**:

- **GET cumulative** : the accumulated values of transaction volume, system cost, chargebacks and signaled fraud;
- **GET timeseries/{value}** : time series for the evolution of the values chargebacks, *global risk score*, lost value, signaled fraud and transaction volume;
- **GET category/{name}** : list of available elements to choose for the filtering attributes (name) : acceptor country, payment channel and instrument country;

Furthermore, each endpoint can be filtered by the transaction attributes to display the data at different detail levels. We pass these attributes as query parameters for the endpoints and represent different dimensions in our fact schema Figure 4.7.

To extract these insights, we focus on the data aggregation capabilities of the MongoDB queries. The queries are composed of a pipeline divided into different data transformation stages. Each query has in common the first stage that filters the data based on the query parameters passed to the endpoint where the merchant identification is mandatory. The query parameters constitute the dimensions available for filter analysis in the dashboard. Below we have a description of the meaning of each:

- **Instrument country**: the country associated to the instrument object for the payment transaction (see details in section 3.1)
- **Acceptor country**: country of the payment acceptor, where the merchant created the payment transaction, e.g. merchant terminal
- **Risk channels**: the available fraud detection services for the merchant
- **Start time**: start date of the period to consider for the data analysis
- **End time**: end date of the period to consider for the data analysis
- **Payment method**: the type of payment method associated with the transaction, e.g. Paypal, Multibanco

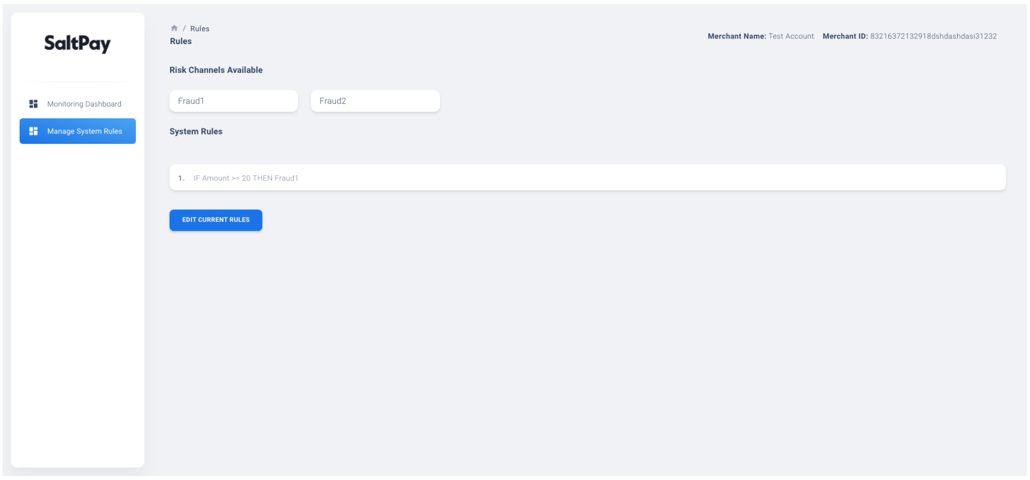
Chapter 5

Results Validation

5.1 End-to-end Testing

This section presents an end-to-end testing scenario of how a customer could benefit from our solution. We will identify the functional requirements defined in section 3.2.1 that are accomplished with each task. We want to highlight the cycle of improvements possible to accomplish that reflects the complete stack of functionalities. The customer will configure the execution of the system, observe the monitoring results and extract conclusions to support adequate changes to the system to respect the business needs.

Step 1: The customer starts with the system configuration. To complete this step, it goes to the risk configuration page and creates the rule *1. if amount > 20 then Fraud1*. As explained in 4.6.1, it selects the "Edit Current Rules" button and opens the edit view, where it creates the wanted rule and saves the changes. The system rule after this change is in Figure 5.1 and respects **F1** and **F4**.


Rules

Merchant Name: Test Account Merchant ID: 83216372132918dshdashdasi31232

Risk Channels Available

Fraud1

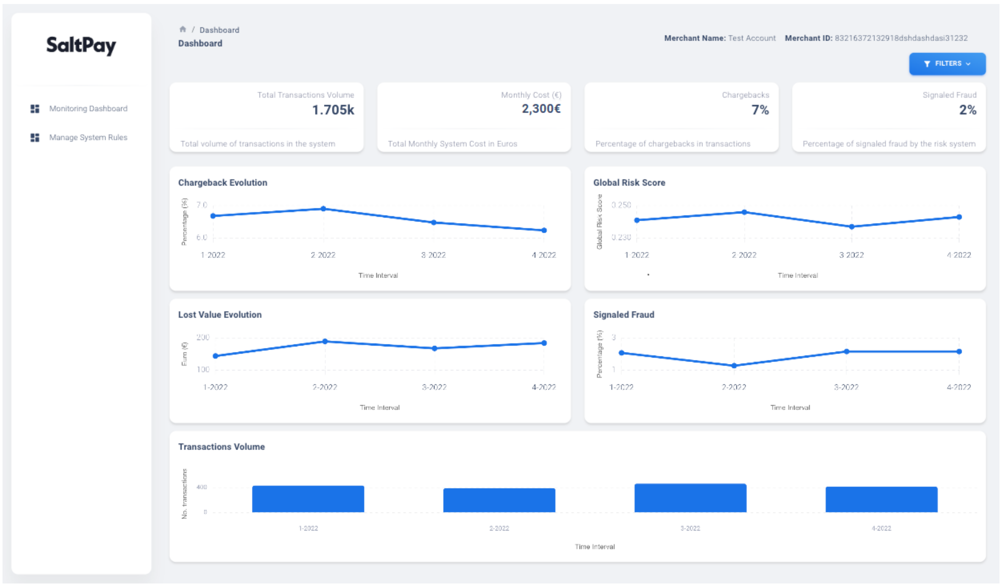
Fraud2

System Rules

1. IF Amount >= 20 THEN Fraud1

Figure 5.1: Configuration of a new rule in the system using the GUI available

Step 2: After the execution of the system for four months, the customer inspects the monitoring data available to understand if there is a need to improve the execution outcomes. As we can see in Figure 5.2 the values of chargeback are around 7%, and the percentage of detected fraud by the system is only 2%, which are not satisfying values for the customer. This step respects **F5**.


SaltPay

Dashboard / Dashboard

Merchant Name: Test Account Merchant ID: 83216372132918dshdashdasi31232

Total Transactions Volume 1.705k — Total volume of transactions in the system

Monthly Cost (€) 2,300€ — Total Monthly System Cost in Euros

Chargebacks 7% — Percentage of chargebacks in transactions

Signaled Fraud 2% — Percentage of signaled fraud by the risk system

Figure 5.2: System monitoring data after the execution of the system for four months

Step 3: After analysing the monitoring data, the merchant wants to change the fraud detection service to improve the percentage of detected fraud and decrease the percentage of chargebacks. This way, after studying better fraud detection services for the market segment, the customer will change the third-party and modify the rules configuration to *if amount > 20 then Fraud2*.

Step 4: After some more months, the customer re-evaluates the system performance and is pleased to observe as in Figure 5.3 that even though the system cost increases, the percentage of fraud detection increases and the percentage of chargebacks lowers.

Step 5: To try to decrease the cost of the tool, the merchant experiments with a serial execution configuration. Thus, the merchant creates the following rules: *1. if amount > 20 then Fraud1* and *2. if Fraud 1 score > 0.4 then Fraud2*. All the transactions go through Fraud1 for a first assessment. If the normalised score of Fraud1 is higher than 0.4, the system uses Fraud2 to get more accurate results. This way, the merchant tries to concede the need for a more accurate service for labelled lower-risk transactions for the first service. In doing so, the merchant tries to decrease the service cost without compromising the risk assessment. The transactions that go through both services respect **F2** and **F3**.

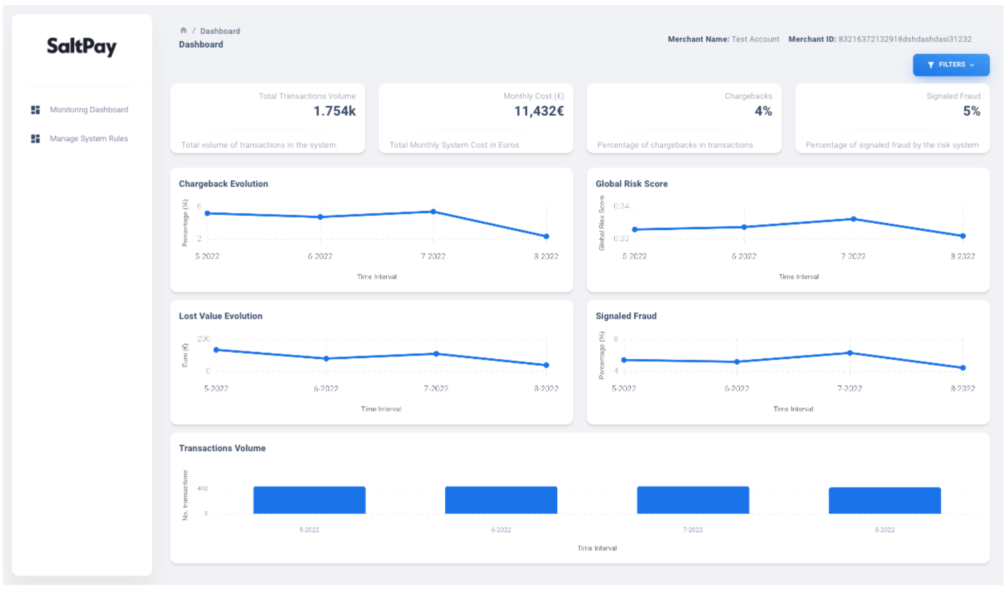


Figure 5.3: System configuration state after applying the new fraud detection service

5.2 Load Testing

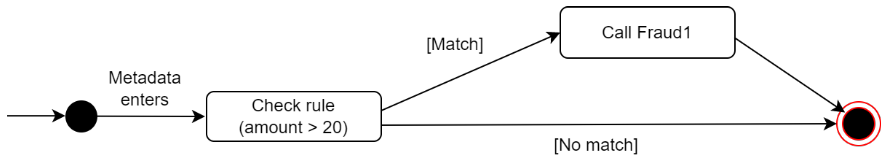
5.2.1 Objectives and methodology

To validate and analyse the performance of our solution, we defined a testing scenario of a possible implementation for a merchant. We executed the test via a script and created about 5200 transactions equally distributed over a year in a local environment. The transaction volume emulates a possible scenario for a real-world merchant's activity and ensures enough transactions to evaluate performance metrics considering the time frame and resources available for this project. To run the Risk module and extract the results, we need two endpoint calls: the first to create a charge and the second to create an instrument. These internal steps of the system company are described in section 3.1.

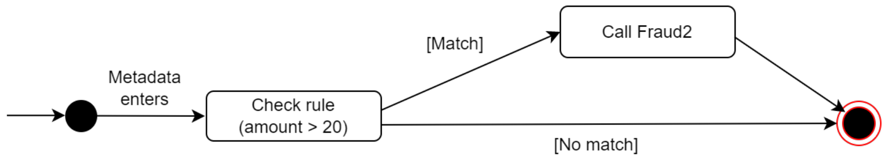
Moreover, for the testing environment, we defined two fraud detection services, Fraud1 and Fraud2, mocked versions of possible service results where Fraud2 represents a service with better detection methods than Fraud1. Even though our initial plan was to test the tool with real fraud detection services, several issues made this situation impossible. The test accounts provided were sandboxes, generated random scores and had time latency discrepant from the endpoints provided to clients. Furthermore, we would have to provide historical data from clients to create an actual account, raising General Data Protection Regulation (GDPR) issues. We also thought of providing anonymised historical data to the services; however, several parameters used for the risk evaluation would be expired or be unavailable for the parameters needed for risk evaluation. To conclude, to properly test with actual fraud detection services, we should involve clients in the testing experience, which would require a timeline unavailable for our project.

Additionally, we determined to block transactions labelled high risk by the system from continuing their flow.

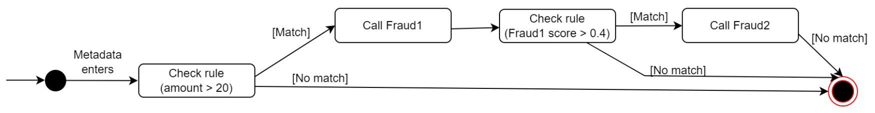
We defined three intervals where the system configurations differ and are presented in Figure 5.4a, Figure 5.4b and Figure 5.4c. In Configuration 1, the merchant opts for a one-rule system that restricts the use of the fraud detection service for transactions with a lower amount than 20€. In Configuration 2, the merchant changes the fraud detection service but maintains the conditions associated with its call. Configuration 3 creates a sequential combination of two fraud detection services where the call to the second one depends on the result score of the first being higher than 0.4. The condition associated with the first service's call is the same as the last configuration.



(a) System configuration 1



(b) System configuration 2



(c) System configuration 3

Figure 5.4: UML state diagram representation of the three applied system configurations

To test the transactions, we identified different testing scenarios based on distinct attributes that we wanted to vary. Therefore, for each configuration, we want to create transactions that follow the traits in Table 5.1. The Risk Level represents the associated level of fraud based on the system global score translation defined in Table 4.1. When transactions reach the end of the payment flow, they can have several outcomes. In our work, we were motivated to represent success, chargeback, and blocked transactions in the Outcome column. A chargeback happens when a customer disputes an item on their account statement, and one of the possible reasons is fraud. For the dashboard, we consider the chargeback element exclusive for fraud-related chargebacks. Blocked transactions are the ones the system considers high risk and, consequently, blocks from continuing their flow.

Notwithstanding, we varied other transaction metrics to have enough variability to illustrate the risk monitoring page in support of our usability testing. However, we will not consider the metrics in Table 5.1 since we did not consider their variation influencing the data generated. The refereed metrics are payment method, instrument country, and acceptor country.

Each transaction can pass by the fraud services or not, since for all configurations, if the transaction amount is lower than 20 euros, then no fraud services will be called.

Table 5.1: Representation of the testing scenarios and the distribution of transaction percentages per configuration

Test Scenario	Uses Fraud Service	Risk Level	Outcome	Configuration 1	Configuration 2	Configuration 3
1	Yes	Low	Success	72.08%	70.02%	69.65%
2	Yes	Low	Chargeback	3.44%	1.33%	2.33%
3	Yes	Medium	Success	13.99%	16.15%	15.07%
4	Yes	Medium	Chargeback	3.38%	1.97%	3.09%
5	Yes	High	Block	1.89%	4.87%	4.50%
6	No	N/A	Success	4.19%	4.57%	4.34%
7	No	N/A	Chargeback	1.03%	1.10%	1.03%

One goal is to divide the test into three intervals with different system configurations to simulate different quality outcomes for the fraud detection result. Thus, we applied different percentages for each testing scenario for each configuration to simulate the amount of detected and undetected fraud. The values are represented in Table 5.1. We want to imply that Configuration 1 had the worst outcome since it had a more significant percentage of undetected fraud (chargebacks) at low and medium-risk levels. Further, it has more transactions with low-level risk despite a lower amount of detected fraud. In addition, as mentioned before, the fraud service Fraud 1 simulates the worst quality service than Fraud 2. Thus, the combination of the percentage division of transaction outcomes and the consistent lower score values despite the same fraud percentage results in the worst outcomes for Configuration 1. For these reasons, Configuration 2 should be considered the best and Configuration 3 in the middle.

5.2.2 Results and discussion

For each transaction, we collected metrics about the total time elapsed within different sections: the process of metadata handling, the third-party execution, and the in-house system. The metadata handling refers to inspecting the requested data to ensure that all the transaction parameters sent are valid within the service. At the same time, it verifies if the transaction sent is already present in the risk database. The third-party execution corresponds to the main component of our developed work, and the in-house system is the rule-engine risk assessment present in the company that we utilise to translate the *global risk score* into an internal system action. All these sections are needed for the regular operation of the developed work, but only the third-party execution was explicitly created for this project. Nevertheless, evaluating the other two is vital to understanding the validity of the solution for the overall time duration because we want to understand if the overall duration respects the stipulated time limits. At the same time, we want to study which components add more overhead to the total time and the influence of the number of fraud services on the time duration of the new solution as shown in Figure 5.8. Regarding the time duration per section, the statistical values are stated in Table 5.2 and will complement our interpretation of the results.

Starting with the total time distribution in Figure 5.5a we observe that most transaction values concentrate on a small range, and we informally identify upper outliers, which should not be considered crucial for our validation procedure. This way, we chose the Interquartile method to formally identify the upper outliers and remove them from the result representations. This decision does not have the intent to discard the existence of outliers. Nonetheless, we can better focus on the remaining data and more easily reach conclusions. The Interquartile method uses the already identified statistical values of the first quarter (1Q) and third quarter (3Q) in Table 5.2 and applies the following steps where the upper fence is the starting limit for considering upper outliers:

1. Calculate the interquartile range (IQR) applying IQR = 3Q – 1Q
2. Calculate upper fence = 3Q + (1.5 * IQR)

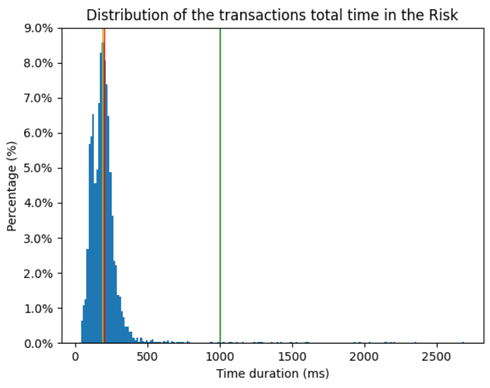
After adequately identifying outliers, we represent the data without them in Figure 5.5b and Figure 5.7a, and we can note that the identification of the most common range of values has improved significantly. We can observe that most data is condensed between 50 and 350ms, which satisfies the time necessities of the solution. Nevertheless, we should remember that we are not considering the real latency of the fraud detection services since we are using mocks in the testing solution. Moreover, even though we have data points that do not respect our time constraints, they were identified as outliers and should not pose a problem to the solution validation.

Table 5.2: Statistical results for the time duration in milliseconds for the total time, metadata extraction, new solution and in-house system

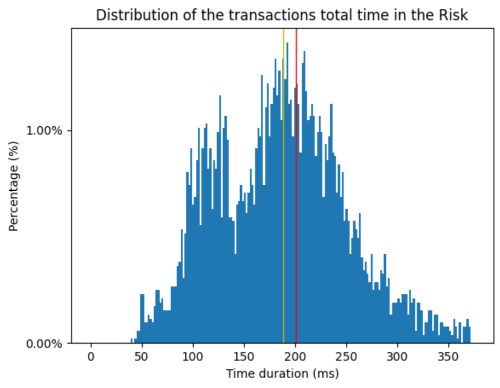
Test Scenario	Average Duration	Max. Duration	Min. Duration	Standard Deviation	1Q	2Q	3Q
Total	200.570	2692	40.469	136.270	136.160	188.592	231.697
Metadata Handling	140.592	2604.296	16.203	120.526	83.051	130.619	163.793
Third-Party Execution	16.588	1798.472	1.176	30.292	11.086	13.847	18.411
In-house System	10.383	146.725	4.141	13.127	5.927	6.808	8.860

Furthermore, comparing Figure 5.6a, Figure 5.7b, Figure 5.6b, Figure 5.7c , Figure 5.6c and Figure 5.7d we conclude that the metadata handling is the element that most contributes to the time duration of the Risk. Thus, in future work, we should explore possible optimisations for this step. Since this step consists of data validation and a database call, we should study how each influences the duration. For the data validation, we should focus on simplifying the process, and for the database, we could study ways to improve the indexation of data and introduce cache to decrease the time duration for frequent computed requests to the database.

From another perspective, we analysed how the number of third-party fraud services influences the time duration inside the new solution section. Presented in Figure 5.8 we recognise that there is a default time duration for the usage of the system regardless of the actual act of calling the fraud services. However, the time significantly increases with the number of providers, which

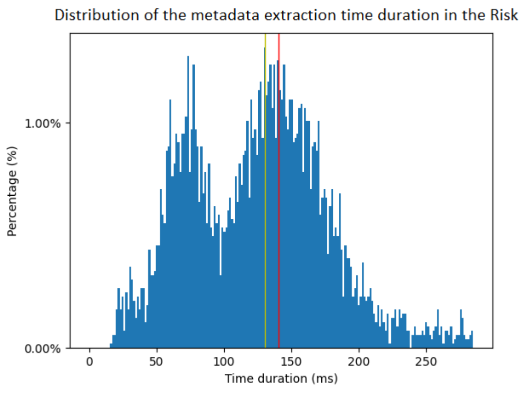


(a) Most transactions respect the time limit stipulations represented by the green line

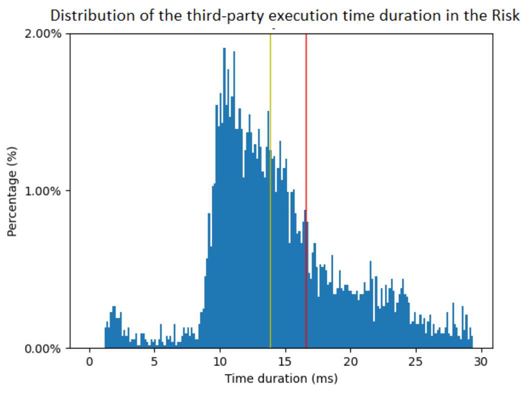


(b) Transactions with filtered outliers respect the time limit stipulations

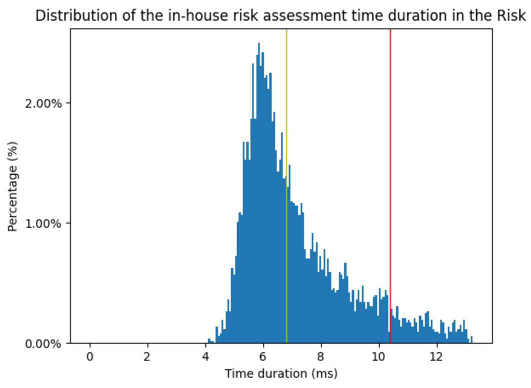
Figure 5.5: Transactions time elapsed distribution in the Risk service; red line represents the mean and yellow the median



(a) Metadata Handling has the most significant impact on the transaction time duration.

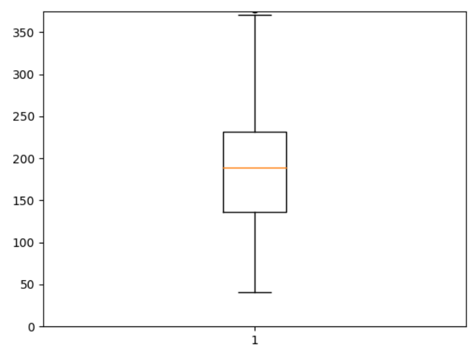


(b) Third-Party Execution

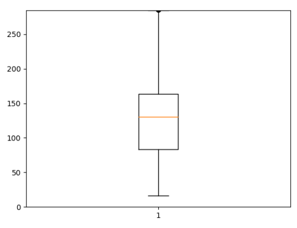


(c) In-house risk assessment

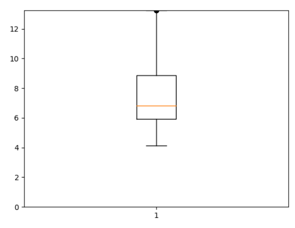
Figure 5.6: Histogram representation of the distribution of time elapsed by the three main procedures inside the Risk service; red line represents the mean and yellow the median



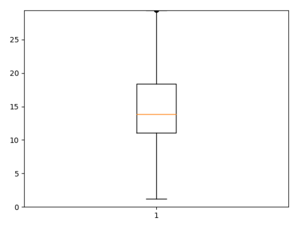
(a) Total Time



(b) Metadata Handling



(c) Third-Party Execution



(d) In-house System

Figure 5.7: Boxplot representation of the distribution of time elapsed by total time and the three main procedures inside the Risk service

represents a limitation to the number of used third parties for each transaction. We can conclude that the third-party execution engine respects the defined time constraints. Unfortunately, we can only make conclusions about the engine because by using mocked services, we can not accurately represent the extra latency created by the external services.

To conclude, the testing results for the Risk service continue to respect the time limits required, representing success for the validity of the developed solution. Analysing the time duration with the increased number of fraud services per transaction could be seen as a constraint to our solution. However, considering a real business scenario, the utility of more than one or two fraud detection services per transaction is shallow. This way, we can not consider a constraint if the purpose of use will not imply more than one or two calls for different fraud detection services in each transaction.

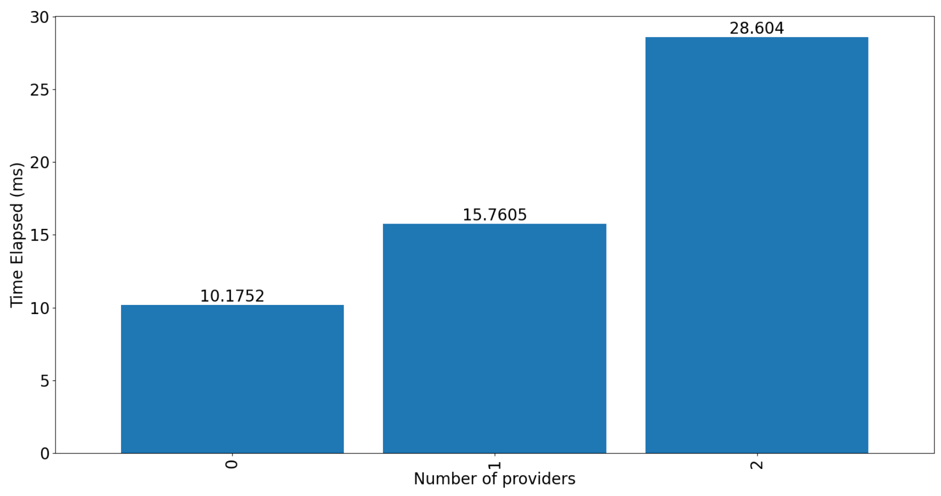


Figure 5.8: Average time elapsed by the number of fraud services called by each transaction

5.3 Usability Testing

This section presents information regarding the usability test carried out for the Risk Management GUI. We divide the work exposition into two parts. In the first one, we explain the methodology applied, and in the second, we discuss the obtained results.

5.3.1 Objectives and Methodology

The usability testing was conducted with the perspective of evaluating the usability quality of the Risk Management GUI. The user should be able to complete the following goals with the interface:

- Manage the rule system
- Correctly interpret the monitoring information of the system execution
- Filter the monitoring information based on transaction characteristics

The target users of this tool are the merchant and customer success agents. However, the most common target is the customer success agent, which performs the requested tasks asked by the merchant. This way, we focused on finding customer success agents to be our testers (identified as tester 1 and tester 2). However, given that the number of customer success agents available was less than five we extended the selection criteria for people with knowledge of the payment processing industry that were not developers. All the testers work daily with a computer and are familiarised with similar GUIs. We recruited males and females with ages ranging from 23 to 36 years old.

We conducted the tests on Zoom with the cameras and microphones on. To execute the tasks, we shared our screen and activated the remote control to allow users to interact with our local project. The sessions were recorded with the appropriate tester's consent to make it easier to analyse the data since only one facilitator was responsible for the tests.

To make it easier to conduct the test and simulate natural conditions of use, we provided our web application with a logged-in fake merchant account with simulated transaction data for a year with the proper characteristics variability.

Each test has a moderator script that should be read to the users to ensure a similar approach for all the tests. We start to explain how the test will proceed and contextualise our work. For the concrete test phase, the methodology applied was to think aloud. We start by asking the tester to look at the homepage and describe what he thinks of it and its functionalities. We apply this method for the risk monitoring and risk configuration pages. After, we ask the user to complete some defined tasks associated with the possible user actions on the page, and when the step is completed, we ask for their opinion of the experience and tested interface. After finishing the test, we ask the user to complete a post-questionnaire based on the system usability scale score (SUS).

Before running the usability tests, we asked one person to run a pilot practice to gain experience and understand the best test approach with the users.

The test artifacts - moderator script, task list, consent form - are present in Appendix C and the SUS post-test questionnaire is present in Figure B.1.

To analyse the results, we defined quantitative and subjective metrics summarised in Table 5.3.

Table 5.3: Subjective and qualitative metrics used to evaluate the GUI usability

Subjective Metrics	Qualitative Metrics
	Number of tasks completed sucessfully
Tester opinions	Time on task
	SUS score

5.3.2 Results and Discussion

We start the result interpretation with the time on task reported in Table 5.4. Each column header has a time reference representing the task time limit to consider it a success. We can observe that the average time duration was respected for all the tasks. One user did not fulfil the time requirements for T1, T6, and T9. However, these values did not influence the success criteria for the average time. Further, they all happened with the same user who was not a customer success agent and had the least experience working with similar user interfaces, which can explain why the tester registered higher response times. This way, we consider that the time duration criteria were successfully achieved.

The remaining qualitative metrics also point out successful validation. All the testers completed the tasks successfully, and the SUS score was considered an above-average result by achieving an average value higher than 68, as presented in Table 5.5. It is also interesting to analyse the results by tester since the three testers with considerably better scores included both customer success agents (tester 1 and tester 2).

To try to better understand the possible factors to improve we present Table 5.6 where we apply to the results of each question the following formula:

- for odd-numbered questions, subtract 1 from the score.
- for even-numbered questions, subtract their value from 5.

With this scale, we have values from 0 to 4, where a higher value corresponds to a higher agreement with the question made and higher satisfaction with the usability factor evaluated by the question. This way, we can observe that one of the questions with a lower average score regards the tool's utility to the user. Since only two of the tester's job functions where the interface would have a direct benefit, we can presume this as the low score average, especially being backed up with the two higher scores from tester1 and tester2. At the same time, another lower score average refers to the integration of functions inside the tool. However, we can observe that the lower score is a direct consequence of the score given by tester5. This way, we should carefully evaluate the opinion segment of tester5 to understand this discrepancy from the remaining testers. After analysing the remaining scores, we can point to other targets for improvements: user confidence, easiness in learning the system and inconsistencies.

Table 5.4: Task time measurement for each tester; the limit to consider success is represented in the header for each task; (X) represent the tests that did not respect the time limit

#	T1 (50s)	T2 (20s)	T3 (10s)	T4 (25s)	T5 (40s)	T6 (40s)	T7 (50s)	T8 (30s)	T9 (80s)
Tester1	30s	7s	3s	6s	20s	25s	32s	10s	17s
Tester2	43s	13s	3s	7s	29s	20s	31s	22s	34s
Tester3	41s	6s	2s	8s	14s	27s	44s	8s	25s
Tester4	49s	5s	3s	7s	28s	27s	39s	14s	33s
Tester5	59s (X)	12s	3s	22s	31s	61s (X)	43s	11s	141s (X)
Average	44.4s	8.6s	2.8s	10s	24.4s	32s	37.8s	13s	50s

Table 5.5: Usability testing SUS scores for each tester and average score

#	Score
Tester 1	87.5
Tester 2	77.5
Tester 3	80
Tester 4	67.5
Tester 5	57.5
Average	74

For the subjective metrics, we asked the testers about their opinion of the experience with the interface. Thus, we summarise their shared assessment in the following points:

Table 5.6: SUS questionnaire score by question for each tester

Tester / Question	Tester 1	Tester2	Tester3	Tester4	Tester5	Average
1	4	3	3	2	1	2.6
2	4	4	4	2	4	3.6
3	4	4	4	4	3	3.8
4	4	4	4	4	4	4
5	4	3	3	3	0	2.6
6	3	3	3	3	4	3.2
7	4	4	4	3	1	3.2
8	4	4	3	3	4	3.6
9	4	3	3	3	2	3
10	4	3	4	3	4	3.6

Tester 1: Tester 1 had initial difficulty understanding the monthly cost element for the monitoring page. In the end, after explaining the meaning, the tester thought that this value was not so relevant to the dashboard and that we should not mix this sensitive content with other execution data. The risk channel concept was not intuitive to understand.

It was also suggested to export the data available to an external document, e.g., CSV. Moreover, the tester considered combining different values of the same filter in a single graph. For the risk configuration page, the tester suggested turning the channel list into clickable elements with more details about the specific channel.

Tester 2: Tester 2 noticed the lack of the default time frame in the monitoring dashboard and the lack of units in the amount value. Like Tester 1, the tester had difficulty understanding the monthly cost parameter. The tester pondered the grey scale utilised in the rule text on the risk configuration page, thinking it could be associated with an inactive state.

The tester suggested some new features for future work: add a filter for specific rules; describe how we calculate the *global risk score* or which are the levels the merchant considers to take action for the transactions.

Tester 3: Like tester 2, tester 3 suggested the description of the *global risk score*. When adding a new rule, the tester considered it strange to appear a new element for adding a new rule from the default view to the edit view. Like Tester 1, the tester considered it relevant to have the list of channels clickable to present more information. The tester also considered the option to edit each rule separately.

Tester 4: Tester 4 took a bit longer than expected to understand that after confirming the deletion of a rule, it should also save the changes on the page to apply them. Based on this, the tester asked for fewer confirmation steps when realising a task.

Tester 5: The last tester also had difficulty understanding the monthly cost element. Like tester 2, the tester noticed the lack of time interval for the default dashboard. The tester suggested switching the number of the month for the name in the graphs and considered that the values in the graph labels should have units. The tester suggested adding the absolute value in the elements with percentage scales.

A common point to all the testers was the lack of description for the elements in the monitoring dashboard. After evaluating all the opinions, we defined a list of improvements for the interface as future work where most of the points are minor usability improvements.

Monitoring page

- Add description for each dashboard component using a tooltip icon; the *global risk score* component should describe the values chosen to block the transactions
- Add the current time interval outside the filter section
- Change the graph labels for the x-axis to present the month identification by name;
- Add units to the label of each value point
- Change the "Monthly Cost" to "Cost" and calculate the value based on the selected period
- Possibility to extract the data to an external document (new functionality)

Risk Configuration Page

- Make the list of possible risk channels into clickable elements where details regarding those fraud services can be found: cost details, start date usage, end date usage, etc.
- Change the grey colour present in the rule text to black
- Remove the intermediate confirmation step for rule deletion

In conclusion, with the results of the metrics defined explored, we can confidently say that the usability test outcome was positive and that the tool meets the required standards for usability. Besides, we identified minor interface problems that should be considered for future work to improve the user experience with our GUI.

Chapter 6

Conclusions

This chapter reflects on the key conclusions and final remarks of the work developed in this dissertation. We divide the content into two sections. In the first, we address how our results were able to support the goals of the investigation and make an evaluation of the contributions associated with our final work. In the second section, we explore open opportunities for future work.

6.1 Achievements and Results

The payment industry is a vast and fast-growing field, especially with the growth of online payment transactions, and many fraudsters notice the phenomenon as an opportunity to explore the system's vulnerabilities. Thus, our proposal intended to create a solution that could diminish the impact of fraudulent transactions on standard merchant payment processing systems.

Concerning the work process, the problem definition and the consequent extensive research provided great insight into our work's business and technological context and allowed us to define the correct path to developing our final solution. Ultimately, we reach a final solution that respects the proposal's primary goals.

We explored and achieved a configurable risk assessment solution based on external fraud detection services. With the implemented service, the customers can have a ready-to-use solution that provides insight to support a decision-making process to improve its performance.

We encountered some challenges to the result validation, given the impossibility of testing actual transaction data with third-party fraud detection services and obtaining real scores for transactions. Nonetheless, the goal of our solution was to provide configuration mechanisms to support the improvement of the fraud accuracy and not the evaluation of the system fraud outcome for a static scenario since we focused on continuous configuration and monitoring mechanisms to improve the system results. Thus, we could still conduct a robust validation of the execution process and extract relevant data to support our work.

With the obtained results, we could confirm the proper functioning of the execution engine with different configuration scenarios. Thus, we revisit the project goals defined in section 1.3.1 and the functional requirements defined in section 3.2.1 to analyse the affirmation above in a more detailed and systematic approach. The testing scenarios proved the success of the creation of working third-party fraud detection services integrations in the company solution flow (**F1** and **G1**); the combination of more than one fraud detection service for the same transaction based on the system rules configuration (**F2**, **G2** and **G4**) with the proper result aggregation into an internal score scale, *global risk score* (**F3**). Furthermore, the extracted statistics related to time responses showed that the system respects the transaction time limits. We did not consider the latency added by real fraud detection services, so we consider (**G6**) partially completed. Furthermore, we were able to identify the portions of work that contribute to a higher transaction time which corresponds to an excellent contribution for future work, detailed in the next section. Furthermore, we accomplished the configuration and system monitoring with endpoints made available (**F6**) and the creation of a GUI for the merchant (**F4**, **F5**, **G4** and **G5**). Regarding the graphical user interface functionalities, we executed a usability test to inspect its quality. The results gave us confidence and validated the developed GUI's usability with a good score. At the same time, and similar to the execution engine validation, it allowed us to create a strategy of improvement for future work.

To finish, we go through the main contributions of our work:

- **Literature review on fraud detection services**: state of the art presents an investigation of the existing fraud detection services and the characterisation of their inputs, outputs and market segmentation (chapter 2, section 2.2.2).
- **Investigation of new fraud detection strategies for payment processing companies**: given the lack of similar solutions, this work provides significant innovation value by exploring a possible new approach to fraud detection and documenting its outcomes.
- **Implementation details**: the document synthesises our solution's implementation details, providing valuable information for replication and proper scientific critical analysis.

6.2 Future work

Although the work developed respected the goal defined in the proposal, there are always points to improve due to lack of time or appropriate circumstances. In the previous chapter, we mentioned possible future improvements for some of the approached contents we will synthesise in this section.

Regarding the execution engine, we could allow the merchant to choose the result aggregation method. In the result validation, we detected the service functionalities which contributed with higher latency to the overall request time and defined a possible approach to improve their performance. For the monitoring data extraction, we identified limitations to the current data model but defined better approaches. Lastly, the usability test resulted in a list of improvements to our GUI to improve the current usability.

Concerning the current solution application in the SaltPay business, upper managers expressed their intention to incorporate it into the existing product even though we did not have time to send the complete work to production. We already have part of the code in production that focuses exclusively on integrating third-parties fraud detection services, even though we do not have any transactions using the integrations. We defined a future work plan to send the final solution to production. Firstly, we will incrementally merge the solution's new features to staging with the appointed improvements. After an experimental phase, we will merge the code into production. Since we are experimenting with a new concept, we will use one or two customers to try our new service, which would comprise about 10% of the total transactions. Furthermore, we will use the prototype rule management interface with our testing client, and in the future, we should incorporate the present features in the company interface.

Regarding the solution's impact on the company, SaltPay has about 80% of the total amount of transactions using their current risk assessment tool. If we focus on the possible candidates inside this group, we could have a vast potential for usage inside the platform.

To conclude, given the continuous increase of payment transactions, especially in the online sector and the consequent fraud, our work can be a consistent and successful approach to detecting fraud readily available to all types of merchants, which could benefit the payment processing industry.

References

[1] Are digital payments set to become mainstream? | thales group. Available at https://www.thalesgroup.com/en/worldwide-digital-identity-and-security/bank-payment/magazine/are-digital-payments-set-become , Accessed last time in July 2022.

[2] Banco best – innovative award winning one-stop-shop website for online financial services| high-yield savings online banking | funds, etfs certificates | stocks, warrants, forex, cfds futures | banco best. ao lado de quem vai à frente. Available at https://www.bancobest.pt/ptg/HomePage , Accessed last time in February 2022.

[3] Classification: Precision and recall | machine learning crash course | google developers. Available at https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall , Accessed last time in February 2022.

[4] Data management and visualization (2021-22) – database and data mining group. Available at https://dbdmg.polito.it/dbdmg_web/index.php/2021/09/13/data-management-and-visualization-2021-22/ , Accessed last time in June 2022.

[5] Druid | frequently asked questions. Available at https://druid.apache.org/faq , Accessed last time in June 2022.

[6] Google pay - learn what the google pay app is how to use it. Available at https://pay.google.com/intl/en_en/about/ , Accessed last time in February 2022.

[7] How credit card payments work | authorize.net. Available at https://www.authorize.net/en-us/resources/how-payments-work.html , Accessed last time in February 2022.

[8] Iso - iec 31010:2019 - risk management — risk assessment techniques. Available at https://www.iso.org/standard/72140.html , Accessed last time in February 2022.

[9] Iso 31000:2018(en), risk management — guidelines. Available at https://www.iso.org/obp/ui#iso:std:iso:31000:ed-2:v1:en , Accessed last time in February 2022.

[10] Mobile payment statistics facts 2022 for marketers. Available at https://www.emizentech.com/blog/mobile-payment-statistics-facts.html , Accessed last time in February 2022.

[11] Online payment fraud statistics: Market summary | infographics. Available at https://www.juniperresearch.com/infographics/online-payment-fraud-statistics , Accessed last time in February 2022.

[12] Online payment processing solution | gocardless. Available at https://gocardless.com/ , Accessed last time in February 2022.

[13] Overview | material dashboard 2 react @ creative tim. Available at https://www.creative-tim.com/learning-lab/react/overview/material-dashboard/ , Accessed last time in February 2022.

[14] Overview • switch docs. Available at https://switchpayments.com/docs/overview#core-concepts , Accessed last time in February 2022.

[15] The payments industry landscape: What does it look like today? - cardknox. Available at https://www.cardknox.com/white-papers/payments-industry-landscape/ , Accessed last time in January 2022.

[16] Pci dss compliance: A guide to securing cardholder data | switch. Available at https://switchpayments.com/learn/6079a1025c98ce001052b40d , Accessed last time in February 2022.

[17] Prepaid cards: simple top-ups and secure payments, even online | nexi. Available at https://www.nexi.it/en/clients/offer/prepaid-cards.html , Accessed last time in February 2022.

[18] Solução de pagamentos - saltpay portugal. Available at https://www.saltpayportugal.pt/ , Accessed last time in February 2022.

[19] Tokenization payment technology guide - adyen. Available at https://www.adyen.com/knowledge-hub/guides/tokenization-payment-technology-guide#how , Accessed last time in February 2022.

[20] World payments report. Available at https://worldpaymentsreport.com/wp-content/uploads/sites/5/2021/12/Top-Trends-2022-in-Payments.pdf , Accessed last time in February 2022.

[21] European Central Bank. Payments statistics: 2020. Available at https://www.ecb.europa.eu/press/pr/stats/paysec/html/ecb.pis2020~5d0ea9dfa5.en.html , Accessed last time in February 2022.

[22] Carol Barnum. *Usability Testing Essentials*. Elsevier Inc., 2011.

[23] Roger Clarke. 21 st bled econference ecollaboration: Overcoming boundaries through multi-channel interaction a risk assessment framework for mobile payments, 2008.

[24] Shaoze Cui, Yanzhang Wang, Yunqiang Yin, T. C.E. Cheng, Dujuan Wang, and Mingyu Zhai. A cluster-based intelligence ensemble learning method for classification problems. *Information Sciences*, 560:386–409, 6 2021.

[25] Juliana de Groot. What is pci compliance? | digital guardian. Available at https://digitalguardian.com/blog/what-pci-compliance , Accessed last time in February 2022.

[26] Alan. Dix. Human-computer interaction. page 834, 2004.

[27] Didier Dubois and Henri Prade. What are fuzzy rules and how to use them. *Fuzzy Sets and Systems*, 84:169–185, 12 1996.

[28] Feedzai. How to detect fraud in less than three milliseconds | feedzai. Available at `https://feedzai.com/blog/how-to-detect-fraud-in-less-than-three-milliseconds/` , Accessed last time in February 2022.

[29] Matteo Golfarelli, Dario Maio, and Stefano Rizzi. The dimensional fact model: A conceptual model for data warehouses. *International Journal of Cooperative Information Systems*, 7:215–247, 1998.

[30] Matteo. Golfarelli and Stefano. Rizzi. Data warehouse : teoria e pratica della progettazione. 2005.

[31] Ralph Kimball and Margy Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley Sons, Inc., USA, 2nd edition, 2002.

[32] Steve. Krug and Mark. Matcho. *Rocket surgery made easy : the do-it-yourself guide to finding and fixing usability problems*. New Riders, 2010.

[33] Rebecca Lake. Buy now, pay later. Available at `https://www.investopedia.com/buy-now-pay-later-5182291` , Accessed last time in February 2022.

[34] Page Laubheimer. Beyond the nps: Measuring perceived usability with the sus, nasa-tlx, and the single ease question after tasks and usability tests. Available at `https://www.nngroup.com/articles/measuring-perceived-usability/` , Accessed last time in February 2022.

[35] Jaehoon Lee, JeongAh Kim, Insook Cho, and Yoon Kim. Integration of workflow and rule engines for clinical decision support services. *Studies in health technology and informatics*, 160:811–5, 2010.

[36] Malgorzata Migut and Marcel Worring. Visual exploration of classification models for risk assessment. *VAST 10 - IEEE Conference on Visual Analytics Science and Technology 2010, Proceedings*, pages 11–18, 2010.

[37] Jakob Nielsen. How many test users in a usability study? Available at `https://www.nngroup.com/articles/how-many-test-users/` , Accessed last time in June 2022.

[38] Jakob Nielsen. *Usability engineering*. Academic Press, 1993.

[39] Jakob Nielsen and Thomas K. Landauer. Mathematical model of the finding of usability problems. *Conference on Human Factors in Computing Systems - Proceedings*, pages 206–213, 1993.

[40] Kyaw Zay Oo. Design and implementation of electronic payment gateway for secure online payment system. *International Journal of Trend in Scientific Research and Development*, 3:1329–1334, 8 2019.

[41] Switch Payments. Overview • switch docs. Available at https://switchpayments.com/docs/overview#core-concepts , Accessed last time in February 2022.

[42] Sazzadur Rahaman, Gang Wang, and Daphne Yao. Danfeng (daphne) yao. 2019. security certification in payment card industry: Testbeds, measurements, and rec-ommendations. *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, page 18.

[43] Marvin Rausand and Stein Haugen. Risk assessment: Theory, methods, and applications. *Risk Assessment: Theory, Methods, and Applications*, pages 1–762, 1 2020.

[44] Allied Market Research. Platform as a service (paas) market statistics | fore-cast - 2030. Available at https://www.alliedmarketresearch.com/platform-as-a-service-market-A06955 , Accessed last time in February 2022.

[45] Lior Rokach. *Ensemble learning : pattern classification using ensemble methods*, volume 85.

[46] Luigi Russis and Fulvio Corno. User evaluation: Usability testing. Available at https://elite.polito.it/files/courses/02JSKOV/2021/slide/12-usability-testing.pdf , Accessed last time in June 2022, 2021.

[47] Ibm Security. Cost of a data breach report 2021. Available at https://www.ibm.com/downloads/cas/OJDVQGR , Accessed last time in February 2022.

[48] Krzysztof Trawinski, Oscar Cordon, Luciano Sanchez, and Arnaud Quirin. A genetic fuzzy linguistic combination method for fuzzy rule-based multiclassifiers. *IEEE Transactions on Fuzzy Systems*, 21:950–965, 2013.

[49] Herausgegeben Von, Wolfgang Semar, and Colin Bolli. Churer schriften zur information-swissenschaft impact of digital payment methods on traditional payment transactions an analysis of the effects on the swiss financial market.

[50] Gang Wang and Jian Ma. A hybrid ensemble approach for enterprise credit risk assessment based on support vector machine. *Expert Systems with Applications*, 39:5325–5331, 4 2012.

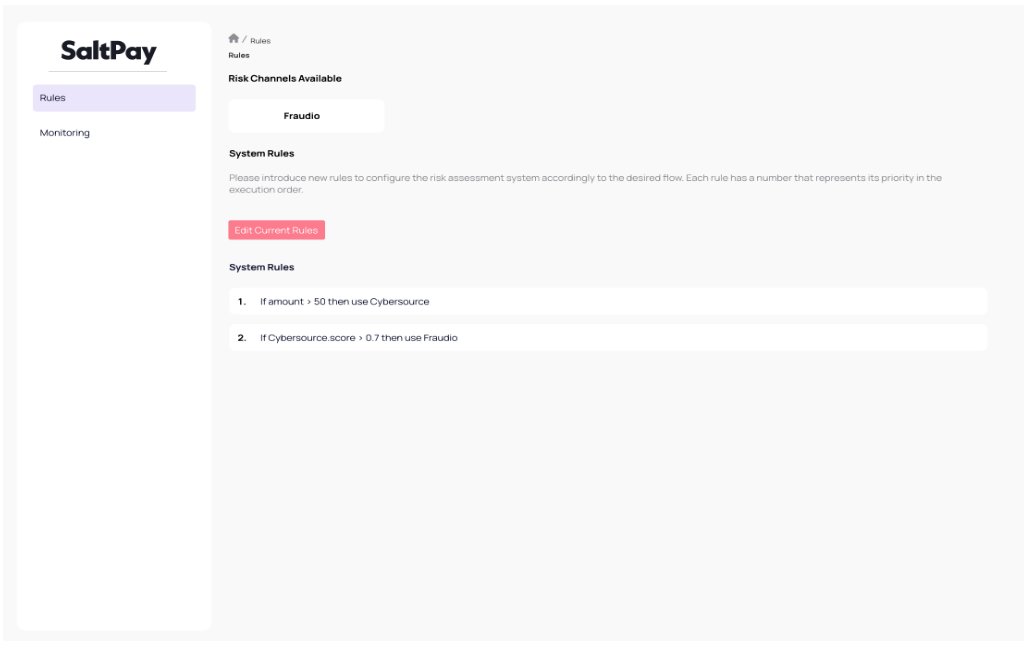
[51] WorldLine. Who are the key players in the payments industry? | worldline canada. Available at https://www.bambora.com/en/ca/learn/payments-industry-players/ , Accessed last time in January 2022.

[52] Lean Yu, Shouyang Wang, and Kin Keung Lai. Credit risk assessment with a multistage neural network ensemble learning approach. *Expert Systems with Applications*, 34:1434–1444, 2 2008.

Appendix A

Risk Management Interface Mockups

A.1 Rule Management UI



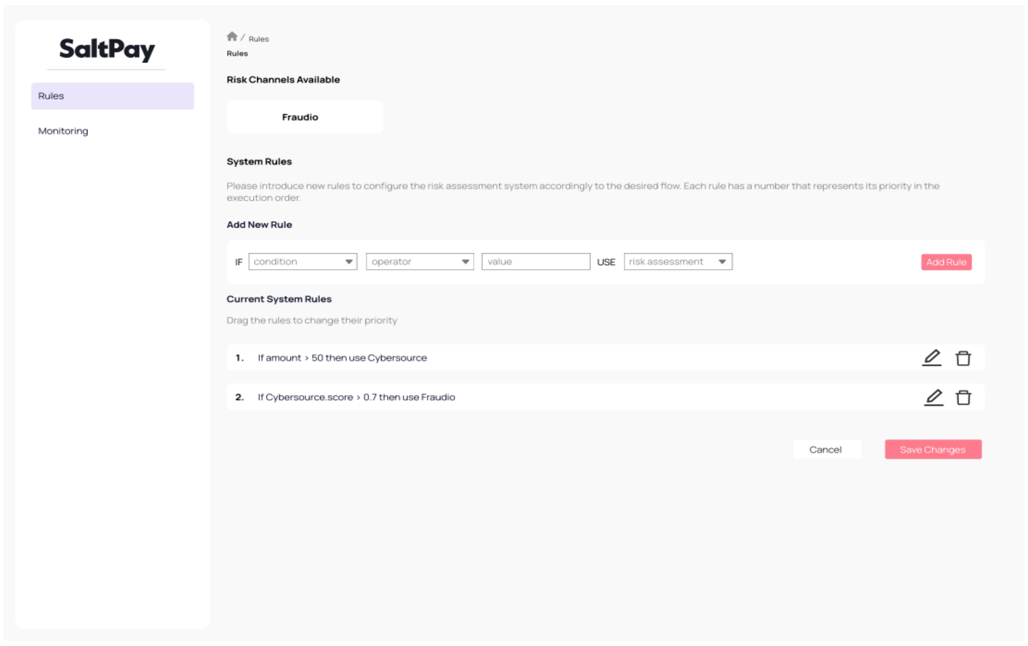
System Rules

Please introduce new rules to configure the risk assessment system accordingly to the desired flow. Each rule has a number that represents its priority in the execution order.

System Rules

1. If amount > 50 then use Cybersource
2. If Cybersource.score > 0.7 then use Fraudio

A.2 Rule Management Edit Mode UI



Risk Channels Available

System Rules

Please introduce new rules to configure the risk assessment system accordingly to the desired flow. Each rule has a number that represents its priority in the execution order.

Add New Rule

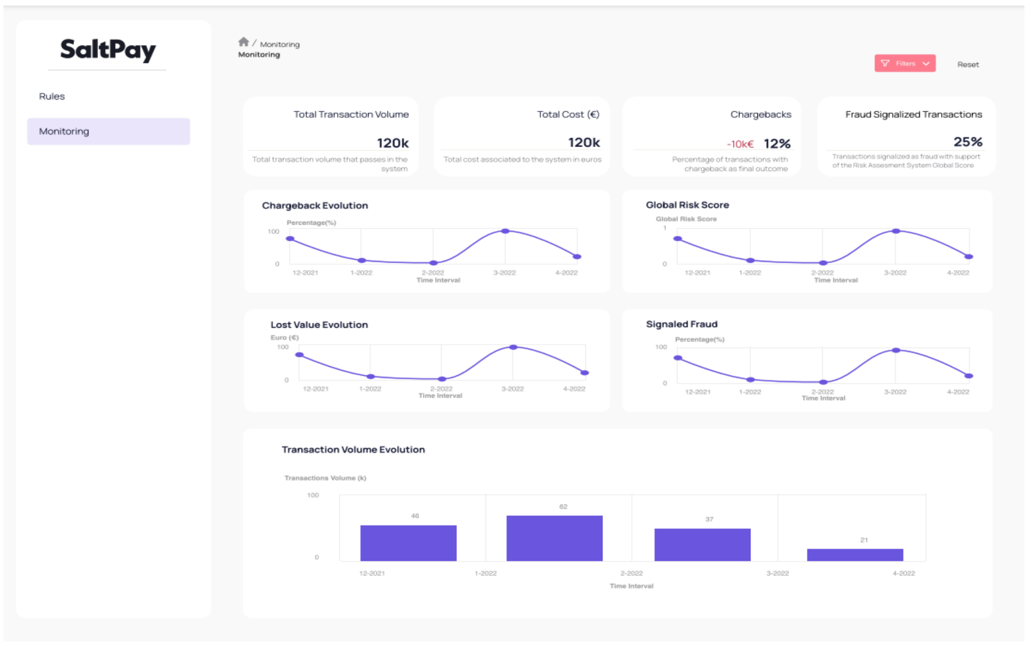
IF condition operator value USE risk assessment

Current System Rules

Drag the rules to change their priority

1. If amount > 50 then use Cybersource
2. If Cybersource.score > 0.7 then use Fraudio

A.3 Monitoring Dashboard UI



Total Transaction Volume	Total Cost (€)	Chargebacks	Fraud Signalized Transactions
120k	120k	-10k€ 12%	25%
Total transaction volume that passes in the system	Total cost associated to the system in euros	Percentage of transactions with chargeback as final outcome	Transactions signalized as fraud with support of the Risk Assesment System Global Score

Transaction Volume Evolution: 46, 62, 37, 21

Appendix B

System Rules Fields and Operators

B.1 Rule Operators

Table B.1: Rule Operators

Operator	Description
>=	Greater or equal
<=	Less than or equal
=	Equal
!=	Different
>	Greater
<	Less than

B.2 Rule Fields

Table B.2: Rule Fields

Field Name	Category	Operators	Example Value
Amount	Amount	>, <, >= , <=	50
Card Bin [1]	Card Characteristics	= , !=	180360
Card Brand	Card Characteristics	= , !=	Visa
Card Type	Card Characteristics	= , !=	credit
Card Bank	Card Characteristics	= , !=	BPI
Charge Type [2]	Transaction characteristics	= , !=	one-time card
Customer IP Country	Location	= , !=	172.22.0.1
Instrument Country	Location	= , !=	PT
Instrument Fingerprint Usages Last Day [3]	Transaction characteristics	>, <, >= , <=	2
Instrument Fingerprint Usage Last Hour	Transaction characteristics	>, <, >= , <=	3
Any Transaction	Unconditional Execution	N/A	N/A
Third-Party-X Score [4]	Score of Previous Fraud Services	>, <, >= , <=	0.4

[1] Bank Identification Number (BIN)

[2] Defines certain charge types, also known as payment methods

[3] The instrument fingerprint identifies the instrument in use (stage of the payment transaction inside the system) without exposing real information.

[4] The normalized score by the system resulting from the third-party fraud detection service named Third-Party-X

Appendix C

Usability Testing

C.1 Moderator Script - Portuguese Version

[Adapted from [32]]

Olá! O meu nome é Ana e hoje vou guiar esta sessão. Antes de começarmos, irei rever algumas informações para ter a certeza que cubro tudo o que é importante saber.

Provavelmente, já tens uma ideia do motivo de estarmos aqui, mas eu irei rever o seu propósito brevemente.

Eu estou a pedir a diferentes pessoas para usarem a ferramenta de gestão de risco de fraude que estou a desenvolver para a minha tese de mestrado, com o objetivo de compreender se tudo está a funcionar da forma o mais correta possível.

Esta ferramenta está destinada aos comerciantes. No entanto, o cenário mais comum será a sua utilização por um operador de *customer success*, de acordo com aquilo que for pedido pelo comerciante.

O sistema de risco desenvolvido tem como objetivo possibilitar aos comerciantes a integração de diversos serviços de deteção de fraude e gerir a sua configuração e monitorização. Para possibilitar a uniformização dos resultados das diversas escalas de risco externas, a ferramenta apresenta uma escala de risco interna chamada *global risk score*. A configuração do sistema passa, por exemplo, por aplicar a restrição: se o valor da transação for superior a 20 euros chamar o serviçoX. A monitorização de resultados tem como objetivo a deteção de problemas e atualização do sistema para a sua melhoria.

A primeira coisa que quero deixar clara é que estamos a testar o site e não a ti. Tu não podes fazer nada de errado nesta sessão. De facto, este é o lugar para não te preocupares acerca de fazeres erros.

Enquanto estiveres a testar o site, vou pedir-te para pensares em voz alta o máximo possível: dizeres para que estás a olhar, o que estás a tentar fazer e o que estás a pensar. Isto será uma ajuda valiosa para o projeto.

Também não te preocupes em ferir sentimentos, porque o objetivo desta sessão é melhorar o site.

Se tiveres alguma dúvida no decurso da sessão, podes perguntar. Eu posso não responder logo, visto que estou interessada em como é que fazes certas ações se estiveres sozinho. Mas se ainda tiveres questões quando acabares, eu tentarei responder. E se precisares de alguma pausa, basta dizeres.

Com a tua permissão, irei gravar esta sessão, que será apenas usada para ajudar a compreender como melhorar o site, e não será do conhecimento de ninguém a não ser de quem está a trabalhar no projeto. Vai ajudar-me a conseguir conduzir a sessão e voltar a rever mais tarde para tirar notas.

Se não tiveres nada a opor, vou pedir para assinares uma permissão de gravação.

Tens alguma questão até agora?

Ok ótimo. [Se não tiveres mais perguntas, vamos começar.]

(Open homepage)

Olhando para esta página, diz o que te parece: como interpretas a sua função, o que consegues ver ou fazer aqui e para que serve? Visualiza e tenta fazer uma descrição sobre a página. Podes dar *scroll*, mas não carregues em nada já.

(Repeat for both pages in the GUI)

Obrigada. Agora vou solicitar-te para tentares fazer umas tarefas específicas. Queria pedir-te que comentasses em voz alta, o que estás a pensar. Começando pela primeira tarefa, irei ler em voz alta, mas também poderás acompanhar pelo ficheiro que te enviei.

Agora que terminamos, tens alguma questão, ou opinião que queiras dar sobre o site?

Desta forma, damos por terminada esta sessão e irei parar a gravação. Obrigada pela tua disponibilidade para apoiares este projeto.

C.2 Moderator Script - English version

[Adapted from [32]]

Hi! My name is Ana, and I'm going to be walking you through this session today. Before we begin, I have some information for you, and I'm going to read it to make sure I cover everything. You probably already have a idea of why we asked you here, but let me go over it again briefly.

We're asking people to try using a interface that we're working on for my master thesis study so we can understand whether it works as intended. We are performing a usability testing that intends to provide a visual tool to support the risk system we designed.

This system aims to integrate fraud detection services in payment processing systems and provide a unique scale to evaluate the risk pointed in the different third parties. We can also define what the conditions of execution of the services are. For example, we can define that we call a service x only when the transaction amount is more significant than 20. The presented tool has two goals: to manage the rules that define the execution of the system and to monitor the global outcomes of the transactions that go through the system. With the monitoring tool, we intend to obtain insight into possible changes and improvements to the system and apply them on the rule system page.

The first thing I want to make clear right away is that we're testing the site, not you. You can't do anything wrong here. In fact, this is probably the one place today where you don't have to worry about making mistakes.

As you use the site, I'm going to ask you as much as possible to try to think out loud: to say what you're looking at, what you're trying to do, and what you're thinking. This will be a big help to us.

Also, please don't worry that you're going to hurt our feelings. We're doing this to improve the site, so we need to hear your honest reactions.

If you have any questions as we go along, ask them. I may not be able to answer them right away, since we're interested in how people do when they don't have someone sitting next to them to help. But if you still have any questions when we're done I'll try to answer them then. And if you need to take a break at any point, just let me know.

With your permission, we're going to record what happens on the screen and our conversation. The recording will only be used to help us figure out how to improve the site, and it won't be seen by anyone except the people working on this project. And it helps me, because I don't have to take as many notes.

If you would, I'm going to ask you to sign a simple permission form for us. It just says that we have your permission to record you, and that the recording will only be seen by the people working on the project.

Do you have any questions so far?

(Open homepage)

First, I'm going to ask you to look at this page and tell me what you make of it: what strikes you about it, whose site you think it is, what you can do here, and what it's for. Just look around and do a little narrative. You can scroll if you want to, but don't click on anything yet.

(Repeat for both pages in the GUI)

Thanks. Now I'm going to ask you to try doing some specific tasks. I'm going to read each one out loud and give you a printed copy. And again, as much as possible, it will help us if you can try to think out loud as you go along.

Now that we finished do you have any question or opinions you would like to give about the tool? This way, we finish this session and I'll stop recording. Thank you so much for your availability to help in this project!

C.3 Task list

#	Descrição das tarefas
T1	O cliente pediu para analisar a percentagem de chargebacks referente a 1 de Janeiro de 2021 até 30 de Abril de 2021. Aplica filtros à dashboard para restringir os dados a este intervalo de tempo e conseguir extrair a análise pedida pelo cliente.
T2	Após esses dados, o cliente pede, no mesmo cenário, para apenas considerar dados referentes a clientes de Portugal. Aplica filtros à dashboard para restringir os dados a clientes de Portugal e conseguir extrair a análise pedida pelo cliente.
T3	Após terminar a tarefa, queres remover os filtros aplicados para voltar à vista default.
T4	Imagina que o cliente quer analisar qual é o custo mensal do canal de risco Fraud2. Filtra os dados da dashboard para o canal de risco Fraud2 para puderes analisar os dados necessários.
T5	Após interpretação dos dados de monitorização, o cliente quer remover a regra com prioridade 2. Indo para a secção de configuração de regras, remove a regra com prioridade 2.
T6	O cliente apercebe se que não compensa fazer análises de fraude para transações até os 50 euros. Edita a regra atual para respeitar esta condição: transações com valor superior a 50 euros passam pelo serviço de fraude.
T7	O cliente pede para acrescentar uma regra do tipo: 'if cardbin = 180360 then Fraud2'. Acrescenta a regra no sistema para respeitar o pedido do cliente.
T8	Após a adição da regra anterior o cliente pede a alteração das prioridades das duas regras existentes. Altera as regras para respeitar o pedido do cliente.
T9	Após várias mudanças no seu negócio, o cliente quer que o sistema de regras funcione em todas as suas transações com o serviço Fraud2. Remove todas as regras existentes e cria uma regra 'if Any transaction then Fraud2'.

Table C.1: Usability task list - Portuguese version

#	Descrição das tarefas
T1	The client asked to analyze the percentage of chargebacks from 1 January 2021 to 30 April 2021. Apply filters to the dashboard to restrict the data to this interval and extract the requested data for the client.
T2	After, the client asks to in the same scenario only consider data of the clientes from Portugal. Apply filters to the dashboard to restrict the data and extract the requested data for the client.
T3	After finishing the task, you want to remove the applied filters and go back to the default values.
T4	Imagine that the merchant wants to analyze the cost for the risk channel Fraud2. Filter the dashboard data for the risk channel Fraud2 to extract the requested data.
T5	After interpreting the monitoring data, the client wants to remove the rule with priority 2. Going to the rule management section, remove the rule with priority 2.
T6	The client understands that it is not worth it to do the analyses to transactions until 50 euros. Edit the current rule to respect the condition: transactions with an amount superior to 50 euros go through the fraud service.
T7	The client asks to add a rule of type : 'if cardbin = 180360 then Fraud2. Add the rule to the system to respect the client request.
T8	After adding the previous version the client asks to alter the priorities of the two existing rules. Alter the rules to respect the client request.
T9	After several changes to the business the clients requires that all the transactions go through the fraud service Fraud2. Remove all the existing rules and add the rule 'if any transaction the Fraud2'

Table C.2: Usability task list - English version

Recording consent form

Thank you for participating in our usability research.

We will be recording your session to allow Ana Margarida Ruivo Loureiro to re-watch to observe your session and benefit from your comments.

Please read the statement below and sign where indicated.

- -

I understand that my usability test session will be recorded.

I grant Ana Margarida Ruivo Loureiro permission to use this recording for internal use only, for the purpose of improving the designs being tested for her master thesis project.

Signature: ______________________________

Date: __________________