UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
PRÓ-REITORIA DE GRADUAÇÃO
DEPARTAMENTO DE ENGENHARIAS E TECNOLOGIA
CURSO BACHARELADO INTERDISCIPLINAR EM TECNOLOGIA DA INFORMAÇÃO

CARLOS GABRIEL FERREIRA

ANÁLISE DE GATEWAYS DE PAGAMENTO: UM ESTUDO SOBRE CUSTOS, FUNCIONALIDADES E DESEMPENHO

PAU DOS FERROS
2025

CARLOS GABRIEL FERREIRA

ANÁLISE DE GATEWAYS DE PAGAMENTO: UM ESTUDO SOBRE CUSTOS, FUNCIONALIDADES E DESEMPENHO

Monografia apresentada à Universidade Federal Rural do Semi-Árido como requisito para obtenção do título de Bacharel em Bacharelado Interdisciplinar em Tecnologia da Informação.

Orientador: Prof. Dr. Walber José Adriano da Silva - UFERSA

PAU DOS FERROS

2025

F368a Ferreira, Carlos Gabriel.
Análise de gateways de pagamento: um estudo sobre custos, funcionalidades e desempenho / Carlos Gabriel Ferreira. - 2025.
48 f. : il.

Orientador: Walber José Adriano da Silva.
Monografia (graduação) - Universidade Federal Rural do Semi-árido, Curso de Tecnologia da Informação, 2025.

1. Gateways de pagamento. 2. Comércio eletrônico. 3. Análise comparativa. 4. Performace e testes. 5. Latência. I. Silva, Walber José Adriano da, orient. II. Título.

Ficha catalográfica elaborada por sistema gerador automático em conformidade com AACR2 e os dados fornecidos pelo) autor(a).
Biblioteca Campus Pau dos Ferros
Bibliotecária: Erlanda Maria Lopes da Silva
CRB: 15/966

O serviço de Geração Automática de Ficha Catalográfica para Trabalhos de Conclusão de Curso (TCC´s) foi desenvolvido pelo Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo (USP) e gentilmente cedido para o Sistema de Bibliotecas da Universidade Federal Rural do Semi-Árido (SISBI-UFERSA), sendo customizado pela Superintendência de Tecnologia da Informação e Comunicação (SUTIC) sob orientação dos bibliotecários da instituição para ser adaptado às necessidades dos alunos dos Cursos de Graduação e Programas de Pós-Graduação da Universidade.

CARLOS GABRIEL FERREIRA

ANÁLISE DE GATEWAYS DE PAGAMENTO: UM ESTUDO SOBRE CUSTOS, FUNCIONALIDADES E DESEMPENHO

Monografia apresentada à Universidade Federal Rural do Semi-Árido como requisito para obtenção do título de Bacharel em Bacharelado Interdisciplinar em Tecnologia da Informação.

Defendida em: 11/12/2025

BANCA EXAMINADORA

Walber José Adriano da Silva, Prof. Dr. (UFERSA)
Presidente

Reudismam Rolim de Sousa, Prof. Dr. (UFERSA)
Membro Examinador

José Ferdinandy Silva Chagas, Prof. Me. (UFERSA)
Membro Examinador

Dedico esse trabalho aos meus pais **Maria José** e **Adalberto Gabriel**, que sempre estiveram ao meu lado, me apoiando e acreditando em mim. Agradeço por tudo.

AGRADECIMENTOS

A Deus, primeiramente, por tudo que fez em minha vida e por ter colocado pessoas em meu caminho que me ajudaram a ser quem eu sou hoje.

Aos meus pais, Adalberto e Maria José, que me escolheram para ser seu filho. Tenho a sorte de tê-los como meus pais. Ao meu pai, meu grande herói, que trabalhou e trabalha para que nunca me faltasse nada, e de fato, nunca faltou. Obrigado por acreditarem em mim e por me guiarem com princípios que levo para a vida. E à minha mãe, que sempre foi um apoio incondicional para mim, me dando força, amor e incentivo em todos os momentos. Meu objetivo é fazer de tudo para dar um futuro bom para vocês, retribuir tudo que fazem por mim. Eu os amo demais.

À minha noiva, Geovana, que sempre esteve ao meu lado ao longo desta graduação, me apoiando e incentivando para que eu não desistisse e corresse atrás dos meus sonhos. Você nunca perdeu a fé em mim e esteve comigo nas alegrias e nos desafios, minha companheira e amiga. Agradeço a Deus por ter você ao meu lado, faz com que todos os meus dias sejam melhores. Obrigado por tudo, meu amor.

Ao meu professor, orientador e amigo, Walber, quero agradecer por ter acreditado no meu potencial, por ter estado ao meu lado e apoiar os meus sonhos. Sem dúvida, você é o melhor professor que já tive. Ao longo desses meses, você se tornou uma referência como pessoa e um grande amigo. Obrigado por tudo o que fez e ainda faz por mim. Você é uma pessoa muito importante pra mim, considero demais.

Aos meus amigos, que sem eles eu não teria conseguido. Vocês fizeram essa caminhada ser mais leve, com varias risadas e histórias que jamais esquecerei.

Felipe Hidequel, uma pessoa muito importante para mim, uma ponte para várias amizades e um irmão que a vida me deu. Seu jeito engraçado com certeza tornou e torna meus dias melhores. Minha gratidão é imensa, pois sem você eu nem sequer teria entrado no curso. Obrigado por tudo te amo demais.

Denys Rafael, uma pessoa incrível com quem sempre pude contar. Você me deu ótimos conselhos ao longo da minha vida e também proporcionou muitos momentos de alegria e risadas, sempre me fazendo rir com suas histórias. Nem tenho palavras suficientes para agradecer. Obrigado por tudo, meu amigo, e por fazer parte da minha vida. Amo demais você.

Abner Gama, meu grande amigo, que considero como um irmão. Me proporciona vários momentos felizes, desde o começo, você esteve comigo e fez parte dessa caminhada.

Colaborou para que eu chegasse até aqui, Obrigado por tudo.

Caio Moisés, meu amigão e irmão. Entramos juntos no curso e, desde então, construímos uma amizade muito forte. Você sempre me inspirou a persistir, acreditar e correr atrás dos meus sonhos. É aquele amigo que eu sei que posso contar sempre.

Luiz Fernando, meu outro irmão que a vida me deu, a sua amizade verdadeira e sua parceria me ajudaram a me tornar melhor, você é uma pessoa especial pra mim que sempre pude contar, agradeço por tudo.

Bruno Paiva, um grande amigo que fiz durante a faculdade e que se tornou uma pessoa extremamente importante pra mim. Desde que nos conhecemos, você sempre trouxe leveza, parceria e boas conversas pra essa jornada. Sou muito grato por tudo com certeza colaborou pra que eu chegasse até aqui.

Agradeço aos meus grandes amigos que fiz ao longo dessa caminhada e que com certeza vou levar pra minha vida **Vladymir Guedes, Renato Alves, Heitor Claudino, Augusto Câmara, Nattan Ferreira Lopes, Vinícius Almeida, Pedro Lucas, Jhoan Fernandes, Kennedy Alves, Artur Santos,** obrigado por tudo "Covardes".

EPÍGRAFE

"Se você não gosta do seu destino, não o aceite. Em vez disso, tenha a coragem para transformá-lo naquilo que você quer que ele seja."

(Naruto Uzumaki)

RESUMO

Escolher um *gateway* de pagamento no Brasil pode ser um verdadeiro desafio para um *e-commerce*, especialmente porque há muitos parâmetros envolvidos e diversas opções disponíveis no mercado. Este trabalho busca realizar uma análise de *gateways* de pagamentos considerando aspectos: qualitativos, comparando taxas e funcionalidades; e, quantitativo, em que foi realizado verificações de desempenho nos ambientes de teste nos *gateways* de pagamento do Mercado Pago e *Stripe*. Após análise qualitativa, Entende-se que não existe o melhor *gateway* de pagamento, mas sim, há uma série de compromissos a serem adotados. Por exemplo, o Mercado Pago, atualmente em 2025, oferece a taxa PIX mais competitiva, enquanto o PagSeguro se destaca por taxas mais atrativas em boletos bancários. Na análise quantitativa, foi verificado o tempo de resposta para criação de sessões de pagamento, tanto em testes sequenciais quanto de forma simultânea. Os resultados medidos demonstram que o Mercado Pago é mais permissivo na quantidade de solicitações que podem ser realizadas, enquanto o *gateway* de pagamento *Stripe* é mais restritivo a tais testes, fornecendo indícios dos mecanismos de segurança vigentes. Portanto, a escolha do *gateway* de pagamento a ser adotado depende do alinhamento dos interesses de negócio do *e-commerce* (elementos qualitativos). E, componentes de interface gráfica com o usuário podem mitigar aspectos de desempenho como animações em carregamentos durante o processo de compra (elementos quantitativos), visando aumentar a taxa de conversão de vendas.

Palavras-chave: gateways de pagamento; comércio eletrônico; análise comparativa; performace e testes; latência

ABSTRACT

Choosing a payment gateway in Brazil can be a real challenge for an e-commerce business, especially because there are many parameters involved and diverse options available on the market. This work aims to analyze payment gateways considering both qualitative aspects, comparing rates and functionalities; and quantitative aspects, where we performed performance checks in test environments for the Mercado Pago and Stripe payment gateways. After qualitative analysis, we concluded that there is no single best payment gateway, but rather a series of compromises to be adopted. For example, Mercado Pago, currently in 2025, offers the most competitive PIX rate, while PagSeguro stands out for more attractive rates on bank slips. In the quantitative analysis, we verified the response time for creating payment sessions, both in sequential and simultaneous tests. The measured results demonstrate that Mercado Pago is more permissive in the number of requests that can be made, while the Stripe payment gateway is more restrictive in such tests, providing evidence of the security mechanisms in place. Therefore, the choice of payment gateway to be adopted depends on the alignment of the e-commerce business interests (qualitative elements). And user interface components can mitigate performance aspects such as animations during loading times in the purchase process (quantitative elements), aiming to increase the sales conversion rate.

Keywords: payment gateways; e-commerce; comparative analysis; performance testing; latency

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Nos últimos anos, o comércio eletrônico no Brasil cresceu de forma acentuada, se transformando em um grande pilar fundamental da economia nacional. Este crescimento foi impulsionado pela ampliação do acesso à Internet, pela popularização dos dispositivos móveis e, de forma decisiva, pela aceleração da digitalização provocada pela pandemia de COVID-19 (NICBR. Núcleo de Informação e Coordenação do Ponto BR, 2024). Esse cenário não apenas atraiu milhões de novos consumidores para o ambiente *online*, mas também democratizou o acesso ao mercado digital, permitindo desde micro, pequenas e médias empresas, pudessem competir de igual pra igual em escala nacional.

Dentro desse ecossistema digital, a etapa de pagamento é um momento crítico e decisivo em compras pela Internet. A escolha de um *gateway* de pagamento até a tecnologia que processa as transações de forma segura tornou-se uma das decisões mais estratégicas para um negócio *online*. Uma experiência de *checkout* fluida, segura e que ofereça as modalidades de pagamento preferidas pelo público consumidor, como o parcelamento pelo cartão de crédito e o PIX, impacta diretamente a taxa de conversão[1], a confiança do cliente e, consequentemente, a sustentabilidade financeira do *e-commerce*.

Por outro lado, a mesma expansão do mercado que resultou grandes oportunidades, também trouxe complexidade. Hoje, os gestores de *e-commerce* se deparam com uma grande quantidade de provedores de *gateways* de pagamento, cada um com diferentes estruturas de taxas, funcionalidades e modelos de integração técnica. O problema central, que motiva esta pesquisa, consiste no fato de que as informações essenciais para essa escolha não estão consolidadas de forma clara e padronizada, sendo frequentemente apresentadas de maneira seletiva e sob uma perspectiva de *marketing*, o que dificulta uma análise comparativa e objetiva.

Diante disso, esta pesquisa se justifica pela sua importância prática, ao buscar oferecer uma análise comparativa que capacite os gestores a fazerem uma escolha mais informada, alinhada às suas necessidades operacionais e financeiras. Para endereçar o problema exposto, o trabalho é norteado pela seguinte pergunta de pesquisa: *dentre alguns dos principais gateways de pagamento disponíveis no mercado brasileiro, como compará-los em relação ao custo, funcionalidades e desempenho em cenários de e-commerce?*

Para responder a hipótese do trabalho, o objetivo geral é realizar um estudo comparativo

[1] Percentual de consumidores que efetivamente realizam a ação de compra, frente aos que tem interesse mas não a realizam

de alguns dos principais *gateways* de pagamento disponíveis no mercado brasileiro, a fim de compilar e apresentar um guia fundamentado que auxilie na escolha da solução que melhor se alinhe às necessidades seja elas estratégicas, técnicas ou financeiras. Para alcançar este fim, foram traçados os seguintes objetivos específicos: investigar a literatura sobre o tema; estabelecer um *framework* de critérios de avaliação; compilar dados por meio de análise documental e testes de desempenho; analisar comparativamente as soluções; e, por fim, organizar e discutir os resultados qualitativos e quantitativos do trabalho.

Este trabalho está dividido em: Capítulo 2 apresenta a revisão da literatura e fundamentação teórica do trabalho. O Capítulo 3 apresenta os resultados da análise comparativa dos *gateways* e a discussão dos resultados. Por fim, no Capítulo 4, Foi apresentado as conclusões do estudo, além de comentar as limitações e sugerir possibilidades para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA E REVISÃO DA LITERATURA

Neste capítulo, apresenta a fundamentação teórica que fornece suporte à presente pesquisa, abordando os conceitos essenciais, o contexto de mercado e as bases metodológicas que orientam a análise comparativa dos gateways de pagamento. O objetivo é criar uma referência sólida para o desenvolvimento da análise nos capítulos seguintes.

2.1 Comércio Eletrônico no Brasil

A evolução do comércio eletrônico no Brasil é marcada por um crescimento acelerado, impulsionado pela ampliação do acesso à Internet e pela popularização de dispositivos móveis. Estudos do início dos anos 2000 já destacavam a intensa utilização de tecnologias da informação para integrar processos e viabilizar os negócios na Era Digital, com foco em sistemas de gestão empresarial (ou *Enterprise Resource Planning* - ERP), sistema de cadeia de suprimentos (*Supply Chain Management* - SCM) e relacionamento com o cliente (*Customer Relationship Management - CRM*) (Albertin, 2002).

Segundo MONTEIRO (2025), a digitalização se acelerou drasticamente nas últimas décadas, tendo a pandemia de COVID-19 como um catalisador decisivo que consolidou os canais digitais como um pilar da economia nacional. Durante esse período, 13 milhões de brasileiros realizaram suas primeiras compras *online*, o que evidencia uma profunda mudança cultural. Essa expansão rápida também trouxe à tona desafios importantes, como a complexidade na logística, carga tributária para comércio eletrônico e, principalmente, a necessidade de assegurar a segurança digital contra fraudes.

A medida que o mercado evoluí, a arquitetura que suporta os sistemas de pagamento também foi impactado por transformações. Várias plataformas de e-commerce eram desenvolvidas sobre uma arquitetura monolítica, caracterizada por ser uma unidade de sistema completa, implantada como um único processo. Esse modelo apresenta desvantagens significativas como o alto acoplamento entre os módulos, a dificuldade de escalar para suportar picos de demanda e a complexidade na manutenção do código (Stradolini, 2021).

Para superar essas limitações, a indústria migrou para a arquitetura de microsserviços, que consiste em serviços implantados de forma independente e modelados em torno de um domínio de negócio específico, como o de pagamentos (Stradolini, 2021). A migração de um módulo de pagamentos para um microsserviço permite que ele seja escalado de forma independente do resto

da aplicação, garantindo maior disponibilidade e resiliência. Em sistemas de altíssimo volume de transações, essa arquitetura é frequentemente aprimorada com uma abordagem orientada a eventos (*Event-Driven Architecture - EDA*), que assegura o processamento de pagamentos em tempo real ou quase real de forma assíncrona, sendo altamente responsiva e escalável (Vangala; Kasimani; Mallidi, 2022).

2.2 Funcionamento do Gateway de Pagamento

No centro das transações de *e-commerce* está o *gateway* de pagamento, definido como um sistema que autoriza e verifica os processos de pagamento, atuando na transação financeira entre a loja virtual, o consumidor e as instituições financeiras. A segurança da informação é fundamental neste processo porque lida com a transmissão de informações sensíveis do cliente até o sistema bancário. Neste, os devidos repasses financeiros são executados entre os clientes e o lojista.

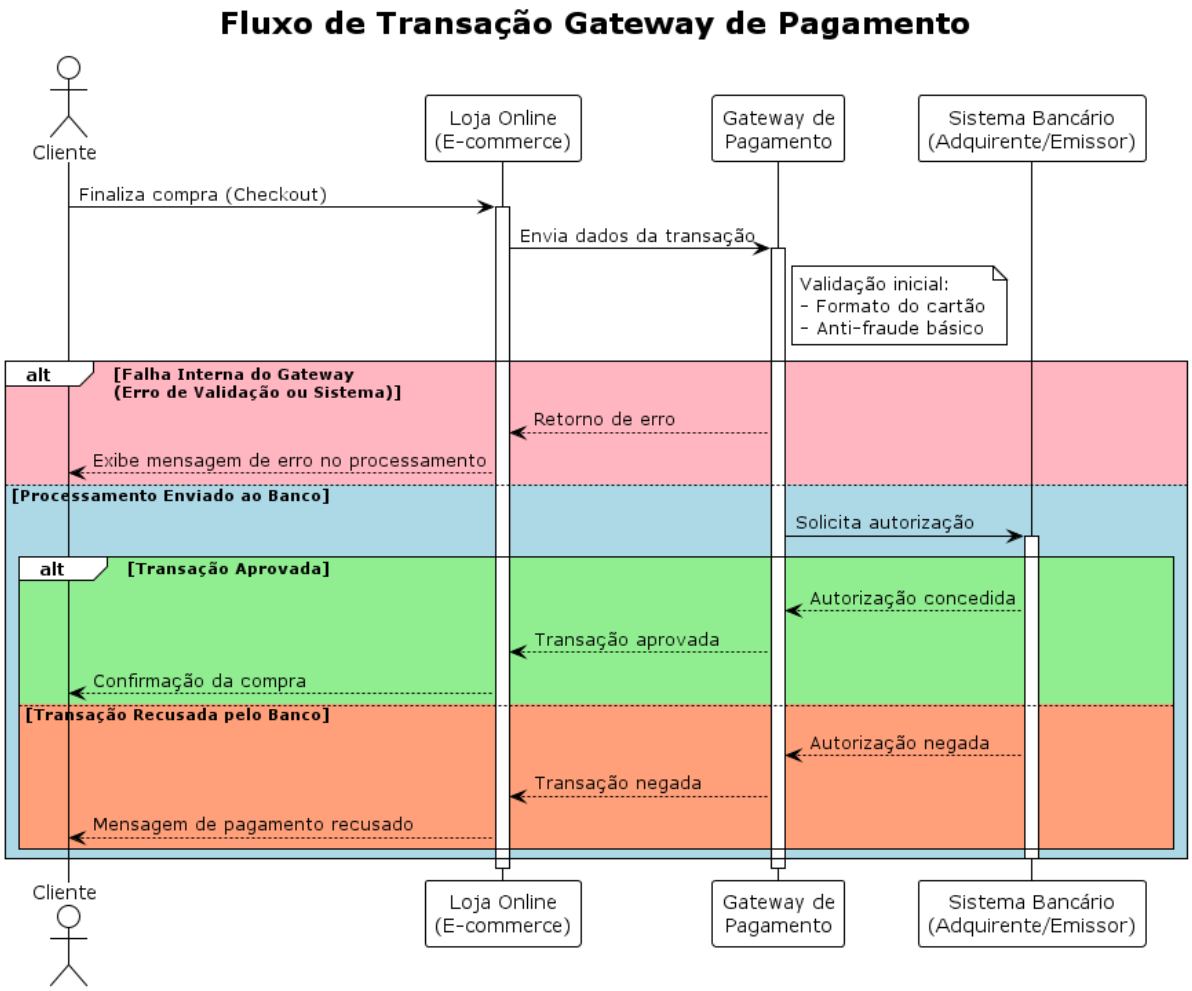
O fluxo de uma transação de um *gateway* é um processo bem definido, que segundo (Hisyam; Manuaba, 2022) ocorre em cinco etapas principais, conforme ilustrado na Figura 1.

O processo se inicia quando o cliente realiza a solicitação de compras no site do *e-commerce*, processo conhecido como *checkout*. A loja, então, contata o *gateway* de pagamento, que se comunica com a rede financeira para validar a transação, checando as informações fornecidas no processo de compra. Após o recebimento das informações vindas do sistema bancário, o *gateway* envia uma resposta de sucesso ou falha de volta para a loja, que por sua vez notifica o cliente.

Além de gerenciar esse fluxo, os *gateways* modernos são essenciais para otimizar a experiência do usuário. A integração entre o *e-commerce* e o *gateway* ocorre, usualmente, via API (*Application Programming Interface*), uma tecnologia que permite a comunicação segura e automatizada entre os sistemas.

No contexto de segurança, os gateways têm um papel importante na redução de riscos, usando sistemas antifraude integrados. Antes mesmo de enviar a transação para o banco, o sistema faz uma análise de risco com a ajuda de algoritmos. Essa análise leva em consideração fatores como a "impressão digital"do dispositivo, a localização pelo IP e o comportamento de compra do usuário. Com esses dados, o gateway pode aprovar a transação, rejeitá-la se houver suspeita de fraude ou encaminhá-la para uma revisão, ajudando o lojista, sem criar obstáculos desnecessários para o cliente legítimo.

Figura 1 – Fluxo geral de uma Transação via Gateway de Pagamento



Fonte: Autoria própria (2025).

2.3 Estratégias de Integração

Um *e-commerce* pode se valer de múltiplas opções de integração com as plataformas de *gateway* de pagamentos. Métodos como "Redirect", ou "Snap", simplificam o processo, onde o usuário é redirecionado a uma página de pagamento externa. Há a possibilidade da integração ser via API, que oferece maior flexibilidade e personalização das páginas de pagamento, permitindo que o *e-commerce* crie uma experiência de *checkout* alinhada ao estilo da página do *e-commerce* (Hisyam; Manuaba, 2022).

A literatura também destaca que as limitações de um único provedor frequentemente levam à necessidade de integrar múltiplos *gateways*. Essa estratégia é adotada para ampliar a oferta de métodos de pagamento, como diferentes carteiras digitais (*e-wallets*), que podem ser exclusivas de cada provedor. Essa flexibilidade possibilita funcionalidades avançadas, como o pagamento dividido (*split payment*), que permite ao cliente pagar uma única compra com

dois métodos de pagamento diferentes (Hisyam; Manuaba, 2022). Estudos de caso práticos demonstram a implementação bem-sucedida de *gateways* renomados como PayPal (Tyagi *et al.*, 2022), Razorpay (Mane *et al.*, 2024) e Dokupay (Husni; Hidayat, 2018).

2.4 Metodologias para uma análise de *gateways* de pagamento

A escolha do *gateway* mais adequado para um *e-commerce* é uma decisão que envolve múltiplos parâmetros a serem selecionados, tanto a nível de negócio quanto a nível tecnológico. O estudo de (Nitto, 2024) apresenta uma análise comparativa de APIs baseada em métricas de capacidade e desempenho. Os critérios de avaliação podem ser divididos em duas categorias principais: requisitos de negócio e não funcionais (ou qualitativos), que verificam aspectos práticos e estratégicos, como a segurança, a estrutura de custos e taxas, a qualidade da documentação de suporte e a usabilidade da plataforma; métricas desempenho: analisam a eficiência técnica do *gateway* e está incluso o volume de dados trafegado, a quantidade de requisições suportadas, a taxa de latência, a taxa de erros e o tempo médio de resposta. (Nitto, 2024).

A aplicação de um *framework* de análise que combine esses critérios quantitativos e qualitativos permite uma avaliação completa, fornecendo uma base sólida para que uma empresa de *e-commerce* possa selecionar o *gateway* que melhor se alinha às suas necessidades operacionais, financeiras e estratégicas.

2.5 Análises Comparativas de APIs e *Gateways*

O trabalho de Nitto (Nitto, 2024) apresenta uma contribuição metodológica relevante, ao propor um *framework* para a análise comparativa de *gateways* de API genéricos, com forte ênfase em métricas de desempenho (latência, taxa de erros, etc.).

Embora este estudo tenha servido de inspiração metodológica para o presente trabalho, ele se diferencia fundamentalmente no escopo. O trabalho de Nitto foca em *gateways* de API de forma ampla, enquanto nessa pesquisa se especializa em *gateways* de **pagamento** voltados para o mercado brasileiro. Além disso, essa pesquisa expande os critérios de análise para incluir fatores de negócio que são cruciais para o *e-commerce* e não foram o foco de Nitto, como as taxas, custos de transação e o suporte a modalidades de pagamento locais, como o PIX e o parcelamento.

2.6 Estudos de Caso de Integração de *Gateways*

Na literatura, encontram-se estudos de caso que exploram a implementação de *gateways* de pagamento para resolver problemas específicos. Hisyam e Manuaba (Hisyam; Manuaba, 2022), por exemplo, focam na integração de múltiplos gateways (como Midtrans e Xendit) para viabilizar a funcionalidade de *split payment* no contexto do mercado da Indonésia.

O referido estudo oferece uma visão valiosa sobre os desafios técnicos da integração de múltiplas APIs. Contudo, o foco deste trabalho é distinto: não se trata de um estudo de caso sobre um único problema de implementação, mas sim de uma análise comparativa prévia (custo, funcionalidades, performance) que visa auxiliar na tomada de decisão de gestores e desenvolvedores no contexto brasileiro, antes mesmo da fase de implementação.

2.7 Pesquisas sobre Arquitetura Interna de Módulos de Pagamento

Outra vertente de pesquisa aborda a arquitetura interna dos sistemas de pagamento. De acordo com Stradolini (2021) apresenta um estudo sobre a migração de um módulo de pagamentos de uma arquitetura monolítica para uma arquitetura de microsserviços.

Esse tipo de pesquisa é focado no desenvolvimento interno de uma solução de *e-commerce*. Já este trabalho segue uma linha diferente, mais voltada para entender, analisar e comparar provedores de *gateways* de pagamento. Dessa forma, o objetivo aqui seria lidar com o problema de escolha de *gateway* disponível no mercado, e não com aspectos mais voltados de arquitetura de software.

2.8 Aspectos de Segurança e Controle de Desempenho em APIs

No desenvolvimento de sistemas distribuídos modernos, especialmente aqueles que lidam com transações financeiras, a atenção não deve se voltar apenas para os Requisitos Funcionais (o que o sistema faz), mas deve ser igualmente rigorosa quanto aos Requisitos Não-Funcionais (RNF), que definem como o sistema se comporta sob certas condições de estresse e risco.

Dentre os diversos atributos de qualidade de software, dois se destacam como críticos para a integração de *gateways* de pagamento: a Segurança (confidencialidade e integridade dos dados) e a Disponibilidade (capacidade de responder sob carga), esta última frequentemente controlada por mecanismos de limitação de taxa.

2.8.1 Segurança e Criptografia (TLS/SSL)

A base da segurança na comunicação entre o *e-commerce* e o *gateway* é o protocolo **TLS (*Transport Layer Security*)**, sucessor do SSL. Este protocolo garante que os dados trafegados sejam criptografados, impedindo que terceiros interceptem informações sensíveis, como números de cartão de crédito (Stallings, 2017).

Embora essencial, a segurança impõe um custo computacional. Para iniciar uma conexão segura, cliente e servidor devem realizar um processo conhecido como *Handshake* TLS. Durante essa etapa, ocorre a troca de chaves criptográficas e a verificação de certificados digitais. A literatura técnica aponta que esse *handshake* adiciona uma latência inicial significativa à primeira requisição de uma conexão, conhecida como *overhead*, que pode chegar a centenas de milissegundos antes que qualquer dado de negócio seja processado (Grigorik, 2013).

2.8.2 Limitação de Taxa (*Rate Limiting*)

Para garantir a disponibilidade do serviço e proteger a infraestrutura contra sobrecargas sejam elas acidentais (picos de acesso em uma promoção) ou maliciosas (ataques de negação de serviço/DDoS) as APIs de pagamento implementam mecanismos de ***Rate Limiting*** (Limitação de Taxa).

O conceito pode ser comparado a uma porta giratória em um banco: ela controla o fluxo de entrada para garantir que o ambiente interno continue operando de forma segura e eficiente. Tecnicamente, o *Rate Limiting* define um "teto"de uso: um número máximo de requisições que um cliente pode realizar dentro de uma janela de tempo específica (por exemplo, 1000 requisições por minuto). Quando esse limite é excedido, o servidor rejeita as novas conexões temporariamente, retornando um código de erro HTTP padronizado, o **429 (*Too Many Requests*)** (Nottingham; Fielding, 2012).

3 METODOLOGIA E DESENVOLVIMENTO

Neste trabalho, propõe-se uma metodologia de pesquisa qualitativa e exploratória, com uma abordagem comparativa, a qual tem como objetivo realizar um estudo sobre os *gateways* de pagamento disponíveis no mercado, analisando desde suas características, funcionalidades, desempenho e taxas. A metodologia é dividida em quatro etapas: primeiro, a definição da abordagem da pesquisa; depois, a seleção dos *gateways* a serem analisados, a definição dos critérios de avaliação e, por último, os procedimentos para coleta e análise dos dados.

3.1 Abordagem da pesquisa

A análise de *gateways* de pagamento foi realizada de duas formas:

- **Análise de documentos:** leitura e estudo das informações que os próprios provedores de *gateway* disponibilizam, como a documentação da API, tabelas de taxas, diretriz de uso e os materiais de apoio para desenvolvedores.
- **Testes práticos:** realização de simulações em ambientes de teste em *sandbox* de cada *gateway* de pagamento selecionado, reproduzindo transações de *e-commerce*. Assim, sendo possível avaliar como um possível sistema de *e-commerce* interage com o *gateway* de pagamento.

3.2 Seleção dos *Gateways* de pagamento

A seleção dos *gateways* de pagamento para este estudo é baseado na sua relevância e popularidade dos provedores do mercado brasileiro. Os critérios para incluir na análise serão:

1. Disponibilidade de um ambiente de *sandbox* acessível para a realização dos testes.
2. Documentação técnica completa e de acesso público.

Com base nesses critérios, foi selecionados os seguintes *gateways* para essa análise comparativa: Mercado Pago [2] e Stripe [3].

3.3 Critérios de Avaliação

Neste trabalho, adota-se uma adaptação do *framework* de análise proposto por Nitto (2024). Essa abordagem se destaca por fundamentar a avaliação técnica em métricas de medição

[2] Site do Mercado Pago: https://www.mercadopago.com.br/developers/pt
[3] Site do Stripe: https://stripe.com/br

de software extraídas da norma internacional **ISO/IEC 25010**, assegurando que os testes e experimentos sejam realizados de forma organizada e confiável. Sendo assim, os critérios foram divididos em duas categorias complementares:

3.3.1 Critérios de Negócio e Funcionalidade

- **Taxas e custos:** Análise da estrutura de preços, incluindo taxas por transação (crédito à vista, parcelado, débito, PIX, boleto), mensalidades e custos de saque.
- **Modalidades de pagamento:** Verificação do suporte a diferentes tipos de cobrança, com foco em:
  - Compras à vista;
  - Compras parceladas (com e sem juros para o cliente);
  - Pagamentos por assinatura;
- **Recursos adicionais:** Disponibilidade de funcionalidades como *split payment*, *link* de pagamento, sistema antifraude e *Checkout Pro*.

3.3.2 Critérios Técnicos e de Performance

Com base na norma ISO/IEC 25010, esta categoria foca em avaliar a eficiência da plataforma em cenários práticos de teste, utilizando as seguintes métricas:

- **Qualidade da API e documentação:** Avaliação da clareza, organização e completude da documentação da API, bem como a facilidade de uso para o desenvolvedor.
- **Modelos de integração:** Análise dos tipos de integração oferecidos, como *Core API* e *Redirect*, conforme discutido na revisão teórica (Hisyam; Manuaba, 2022).
- **Desempenho em testes:** Medição de indicadores de desempenho a partir dos testes no ambiente de *sandbox*, como:
  - **Tempo de resposta:** Tempo médio para a confirmação de uma transação;
  - **Taxa de sucesso/erro:** Confiabilidade do serviço em processar as transações sem falhas e de forma segura.

3.4 Procedimentos de coleta e análise de dados

O estudo será conduzido seguindo os seguintes procedimentos:

1. **Coleta de dados documentais:** As informações sobre taxas e funcionalidades de cada

gateway serão coletadas de seus respectivos sites oficiais e consolidadas em uma tabela comparativa.

2. **Execução dos testes:** Serão criadas contas de desenvolvedor em cada plataforma para acessar o ambiente de *sandbox*. Nesses ambientes, serão simulados cenários de compra comuns em um *e-commerce* para avaliar os critérios técnicos.
3. **Criação de esquemas visuais:** Para cada *gateway* de pagamento, será elaborado um diagrama de fluxo que ilustre o processo da transação, desde o *checkout* na loja até a confirmação do pagamento, facilitando a visualização comparativa.
4. **Análise e Discussão dos Resultados:** Os dados coletados serão analisados de forma comparativa, cruzando as informações de custos, funcionalidades e desempenho.

3.5 Análise do *gateway* de pagamento do Mercado Pago

O Mercado Pago é um dos *gateways* mais populares e utilizados no Brasil, oferecendo um ecossistema completo de soluções financeiras. Para a integração com *e-commerce*, um dos seus principais produtos é o *Checkout Pro*, que funciona como um modelo de integração que redireciona o cliente para um ambiente seguro do mercado pago para concluir sua compra.

O funcionamento dessa integração é centrado no conceito de *Preferência de Pagamento* (*Payment Preference*). A preferência funciona como um objeto de dados, criado via API, que contém todas as informações sobre uma transação específica, como os itens do carrinho, os dados do comprador, as URLs de redirecionamento em casos de sucesso ou falha e as configurações de pagamento. Quando criada a preferência, o *e-commerce* recebe uma URL de pagamento única para direcionar o cliente.

A integração do Mercado Pago baseia-se em um modelo de segurança que divide as responsabilidades entre três partes principais: o *Frontend* (lado do cliente), o *Backend* (servidor da loja) e os servidores do Mercado Pago. A forma de comunicação do *Frontend* utiliza uma chave pública (*Public Key*) apenas para identificação e garantir a confidencialidade das informações, enquanto as operações críticas, que criam a transação, devem ser autenticadas e validadas no *Backend* com uma chave privada (*Access Token*).

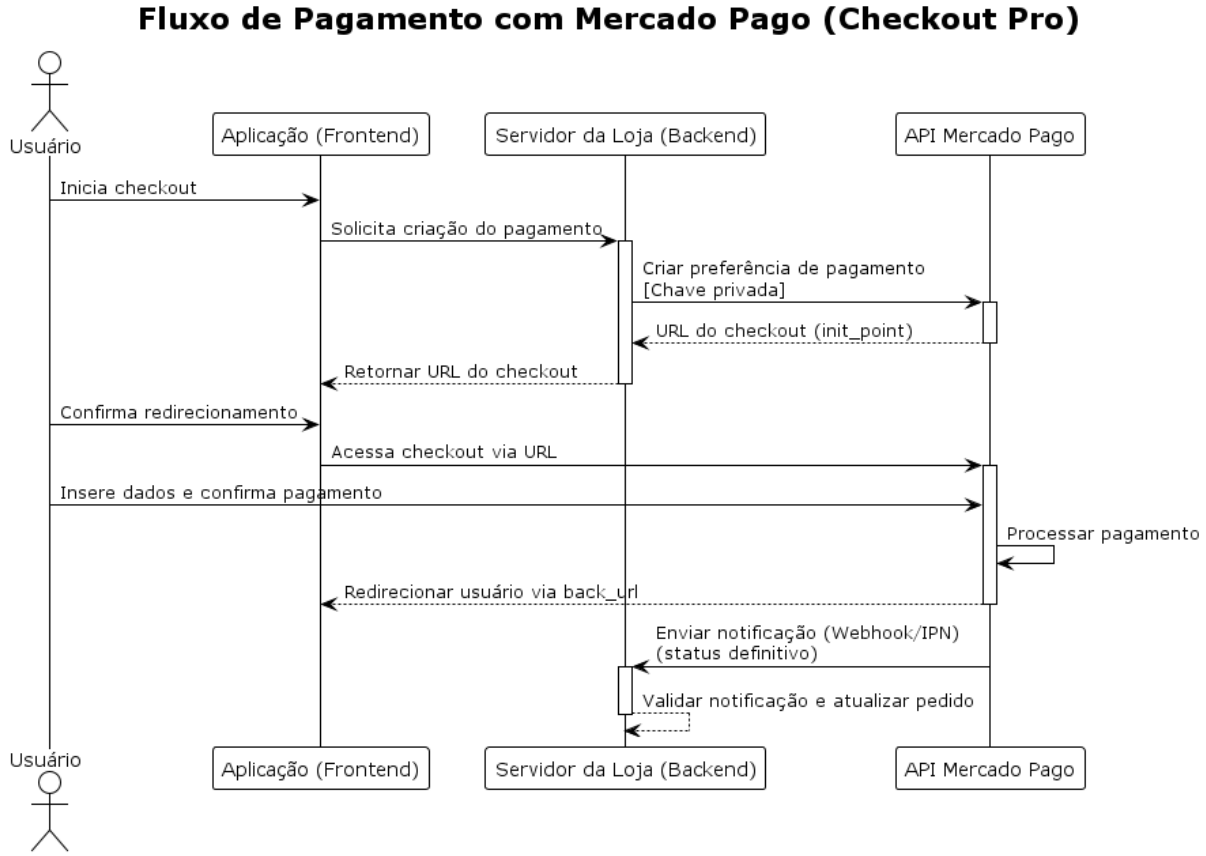
Nesse fluxo, é crucial que a criação da preferência ocorra no *Backend*. O servidor da loja deve primeiro validar a consistência dos dados da compra como os itens do carrinho e, principalmente, os preços com base em seu próprio banco de dados. Se essa verificação de segurança não for feita, pode ocorrer que um usuário, agindo de má fé, podendo interceptar a

requisição no navegador alterando o preço de um produto de por exemplo R$ 1.000,00 para R$ 1,00 antes de enviá-lo ao *gateway*, causando prejuízo financeiro direto para a loja.

Somente após essa validação interna, o *Backend* cria a preferência. Ela funciona como um *"ticket"* de pagamento, gerando um identificador único para essa transação específica. O *e-commerce*, então, recebe a URL de pagamento associada a esse identificador e a envia para o *frontend*, para que o cliente possa realizar o pagamento com segurança.

A Figura 2 ilustra como funciona este processo completo, desde a criação da preferência no servidor até a confirmação final do pagamento.

Figura 2 – Fluxo de Pagamento com Preferência (Checkout Pro) - Mercado Pago



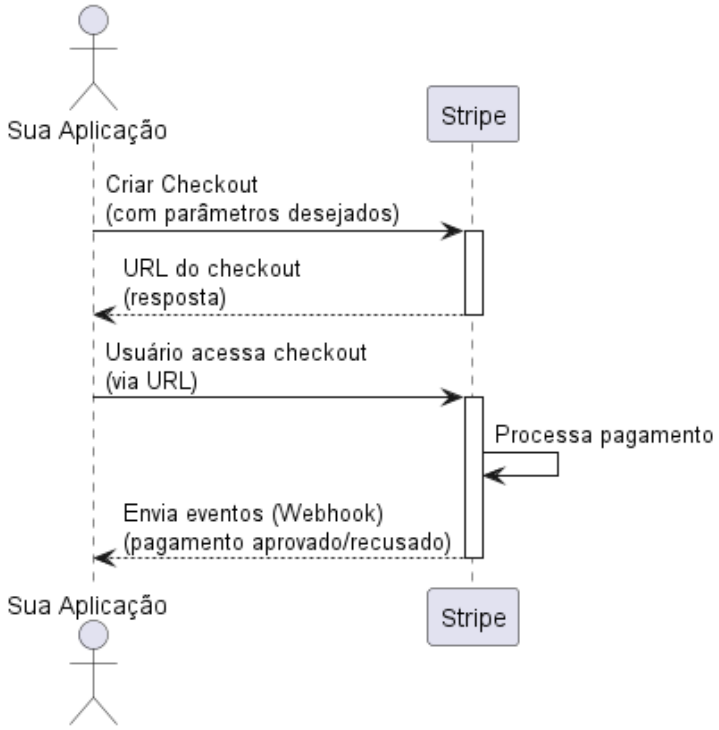
Fonte: Autoria própria (2025).

Esta forma de integração fornece um ambiente seguro, porque os dados do cartão de crédito são digitados diretamente no ambiente do Mercado Pago. A confirmação da transação é feita de forma assíncrona, por meio de uma notificação chamada *Webhook*, garantindo que o *backend* da loja seja notificado sobre o *status* do pagamento (aprovado ou recusado) de servidor do Mercado Pago para servidor do *e-commerce*, conforme a documentação (Mercado-Pago, 2025a).

3.6 Análise do *gateway* de pagamento do Stripe

O Stripe é um *gateway* amplamente utilizado e com forte presença mundial. Pode-se destacar a documentação objetiva e simplifcada, voltada a experiência do desenvolvedor. O mecanismo de transação da Stripe funciona da seguinte forma: o usuário interage com a aplicação do *e-commerce* (Sua Aplicação, Figura 3), depois cria um *checkout* de acordo com suas preferências de compra e em seguida, a Stripe gera uma intenção de pagamento, que é enviada para o usuário. Por fim, o usuário é encaminhado para a área de pagamento em que insere os dados sensíveis para realização da transação financeira. O resultado da operação, seja ela bem sucedida ou não, é encaminhado para o *e-commerce* via chamada *WebHook*, conforme ilustrado na Figura 3.

Figura 3 – Fluxo (Checkout) - Stripe



Fonte: Autoria própria (2025)

A Stripe segue a arquitetura de pagamento centrada no objeto *Payment Intent* (Intenção de Pagamento), esse objeto funciona de forma similar à “Preferência” do Mercado Pago, no sentido de que é um *"ticket"* criado no *backend* que define o valor final da transação. No entanto, sua principal finalidade não é direcionar o cliente, mas sim habilitar um *checkout* transparente e seguro diretamente no *Frontend* da loja, através de uma ferramenta chamada *Stripe Elements*.

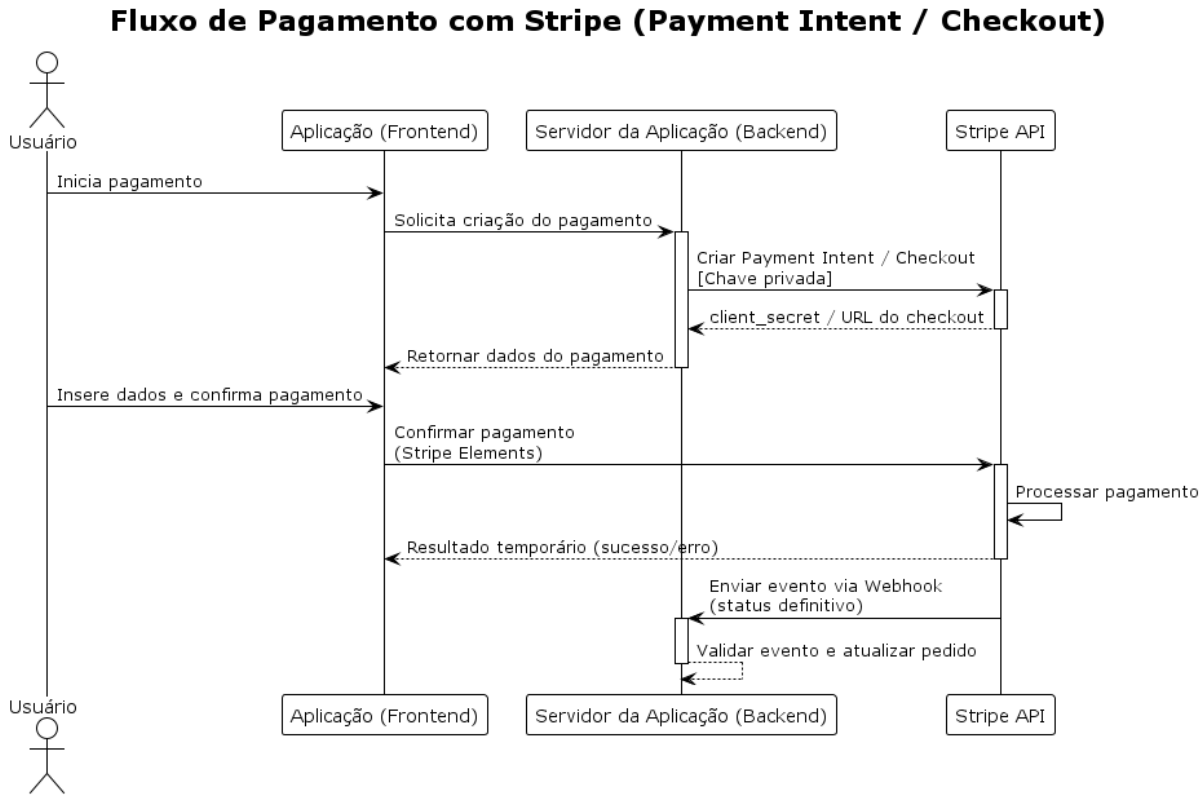
Sobre a segurança, o fluxo de pagamento seguro da Stripe se baseia em separação de

chaves e na criação da *Payment Intent* no lado do servidor:

1. **Criação do *Payment Intent*:** o *backend* do *e-commerce* é responsável por validar as informações de compra do usuário (adicionada no carrinho), como preços, quantidade de itens, etc. Após a validação, é realizada uma chamada à API da Stripe com uma Chave Privada para criar um *"Payment Intent"*, especificando o valor e a moeda.
2. **Recebimento do *Client Secret*:** A API da Stripe retornar o objeto "Payment Intent", que contém um *token* temporário chamado *client secret*. Este *secret* é a "chave" que autoriza o *frontend* a finalizar o pagamento após as infomações estarem de acordo como definido no *backend*.
3. **Inicialização do *Stripe Elements*:** O *backend* envia apenas o *client secret* para o *frontend*. A página de *checkout*, então, usa esse *secret* junto com a Chave Pública da Stripe para inicializar o *Stripe Elements* um formulário de cartão de crédito seguro, em formato de *iframe*, que se permite se adequar ao *design* da loja *on-line*.
4. **Confirmação do Pagamento:** O cliente insere os dados do cartão diretamente no *Stripe Element*. Esses dados são enviados diretamente para a Stripe. O *frontend* então chama a função `stripe.confirmPayment()`, usando o *client secret*, para finalizar a transação.
5. **Notificação via *Webhook*:** Assim como o Mercado Pago, a confirmação do pagamento é feita de servidor para servidor, através de um *Webhook*. O servidor da loja recebe o evento `payment_intent.succeeded`, valida e atualiza o *status* do pedido no banco de dados.

Este fluxo, mostrado na Figura 4, é muito utilizado por ser seguro e oferecer uma boa experiência para o usuário. Isso acontece porque o preço é conferido no *backend*, enquanto o pagamento acontece de forma rápida e direta no próprio site (*frontend*), sem redirecionamentos. (Stripe, 2025b).

Figura 4 – Fluxo de Pagamento com Payment Intent (Stripe Elements)



Fonte: Autoria própria (2025).

4 RESULTADOS

Neste capítulo, vamos apresentar os resultados da pesquisa, que foram obtidos tanto por meio da análise de documentos, disponibilizados pelos *gateways* de pagamento, quanto pelos testes práticos descritos na metodologia. A análise foi feita em duas etapas: uma qualitativa, onde compararam-se os critérios de negócio e funcionalidades dos três *gateways* de pagamento escolhidos; e uma quantitativa, que envolveu os testes de desempenho dos *gateways*, realizados em ambientes específicos de teste para esse propósito.

4.1 Análise qualitativa: negócio e funcionalidades

Na primeira fase da análise, realizamos uma investigação documental para entender as características de negócio de cada provedor. A Tabela 1 reúne os dados sobre taxas, prazos e principais funcionalidades, oferecendo um panorama comparativo dos custos e serviços disponíveis.

Tabela 1 – Análise qualitativa: critérios de avaliação relacionados ao negócio e funcionalidades. Valores de novembro de 2025

Categoria	Critério de Avaliação	Mercado Pago	Stripe	PagSeguro
Critérios de Negócio e Funcionalidade				
Taxas e Custos	Taxa - PIX	0,99% (Mercado-Pago, 2025b)	1,19% (Stripe, 2025a)	1,89% (PagBank, 2025)
	Taxa - Boleto	R$ 3,49 (Mercado-Pago, 2025b)	R$ 3,45 (Stripe, 2025a)	A partir de R$ 1,89
	Taxa - Crédito à Vista	3,98% a 4,98% (Mercado-Pago, 2025b)	3,99% + R$ 0,39 (Stripe, 2025a)	3,99% a 4,99% + R$ 0,40 (PagBank, 2025)
	Taxa - Crédito Parcelado	Taxa base + Custo/parcela (Mercado-Pago, 2025b)	A configurar (Stripe, 2025a)	A configurar (PagBank, 2025)
	Prazo de Recebimento	Na hora, 14 ou 30 dias (Mercado-Pago, 2025b)	Padrão D+2 (variável) (Stripe, 2025a)	Na hora, 14 ou 30 dias (PagBank, 2025)
Funcionalidades	Suporte a Parcelamento	Sim, até 12x (Mercado-Pago, 2025b)	Sim (Stripe, 2025a)	Sim (PagBank, 2025)
	Pagamentos Recorrentes	Sim (Assinaturas)	Sim (Billing) (Stripe, 2025a)	Sim (Planos e Assinaturas)
	Checkout Transparente	Sim (Checkout Pro) (Mercado-Pago, 2025b)	Sim (Elements) (Stripe, 2025a)	Sim (PagBank, 2025)
	Split de Pagamento	Sim (Split e Roteamento)	Sim (Connect) (Stripe, 2025a)	Sim
	Sistema Antifraude	Incluso (Mercado-Pago, 2025b)	Incluso (Radar) (Stripe, 2025a)	Incluso (PagBank, 2025)

Fonte: Adaptado de (Mercado-Pago, 2025b), (Stripe, 2025a) e (PagBank, 2025).

A Tabela 1 mostra alguns *trade-offs* importantes. Por exemplo, o PagSeguro tem a menor taxa para emissão de boleto, enquanto o Mercado Pago oferece a tarifa mais competitiva para PIX. Já a Stripe se destaca pela força das suas ferramentas para desenvolvedores (conforme será visto na análise técnica), embora sua taxa de PIX seja a mais alta entre os três.

Além das taxas, a análise das funcionalidades revela que os três *gateways* oferecem um conjunto robusto de ferramentas essenciais para o mercado brasileiro. Conforme apresentado na Tabela 1, todos suportam parcelamento, uma funcionalidade crítica para o varejo nacional, e oferecem soluções de *Checkout* Transparente (como o *Checkout Pro* do Mercado Pago e o *Elements* da Stripe), que permitem ao cliente finalizar a compra sem sair do ambiente da loja, aumentando a conversão.

Um diferencial importante observado é o suporte nativo a Pagamentos Recorrentes e *Split* de Pagamento em todas as plataformas analisadas. A Stripe se destaca com o produto *Stripe Connect* para *split* de pagamentos e o *Stripe Billing* para recorrência, ferramentas reconhecidas pela flexibilidade na gestão de modelos de negócio complexos, como *marketplaces* e *Software as a Service* (SaaS). O Mercado Pago e o PagSeguro também oferecem essas funcionalidades, mas com opções pensadas para uma integração mais fácil e rápida.

Por fim, no quesito segurança, todos os provedores incluem sistemas antifraude integrados. A Stripe utiliza o *Radar*, que aplica *machine learning* em sua vasta base global de dados para detectar fraudes, enquanto Mercado Pago e PagSeguro utilizam soluções proprietárias ajustadas ao perfil de risco do consumidor brasileiro. Portanto, sob a ótica das funcionalidades, a escolha dependerá menos da "existência" do recurso (já que todos possuem) e mais da qualidade da documentação e da facilidade de implementação de cada ferramenta específica.

4.2 Análise do mercado pago: desempenho em testes

Além da análise qualitativa e da avaliação documental, também foram feitos testes práticos de desempenho para medir e comparar a eficiência técnica das APIs. O objetivo foi obter métricas objetivas, como tempo de resposta e confiabilidade, nos ambientes de *sandbox* do Mercado Pago e Stripe. É importante resaltar que o PagSeguro não foi incluído nesses testes devido o tempo disponível neste trabalho e, também, porque para a liberação do ambiente de testes do PagSeguro se mostrou mais burocrático. Portanto, a análise quantitativa de desempenho será apenas entre o Mercado Pago e a Stripe, com o PagSeguro apenas incluso na avaliação qualitativa (conforme mostrado na Tabela 1) e também na análise dos fluxos de integração.

4.2.1 Teste 1: Execução Sequencial (Mercado Pago)

O primeiro teste consistiu em uma execução sequencial (*serial*), simulando o cenário de criação de uma **Preferência de Pagamento** objeto que encapsula todos os dados da transação (como itens e valores) para gerar o *link* de *checkout*. O teste operou como uma fila de pedidos, em que uma nova requisição só é disparada após a conclusão da anterior.

Durante os testes iniciais, foi notado que a primeira requisição da série apresentava uma latência bastante alta, superior a 500 ms, enquanto as próximas se estabilizavam em torno de 103 ms. Para entender melhor a causa dessa demora, foi considerada a hipótese de que o tempo

de resolução do DNS *(Domain Name System)* pudesse estar afetando o resultado na primeira tentativa.

Para testar essa ideia, foi configurado o ambiente de teste em nuvem controlado. O *script* de teste foi executado a partir de uma instância dedicada com as seguintes especificações:

- **Infraestrutura:** Amazon Web Services (AWS) - EC2.
- **Região:** `us-east-1` (N. Virginia, EUA), escolhida por sua alta conectividade.
- **Instância:** Tipo `t3.micro` (2 vCPUs, 1 GiB Memória).
- **Sistema Operacional:** Ubuntu Server 24.04 LTS.
- **Rede:** Endereço IPv4 Público estático, garantindo conectividade com a Internet.

Mesmo após essa alteração, a latência alta na primeira requisição continuou, o levou a concluir que esse *overhead* inicial não está relacionado ao DNS. Na verdade, ele é causado pelo processo de *handshake* TCP/TLS (SSL), conforme evidenciado posteriormente na análise de pacotes (Figura 11), que demonstra a abertura e fechamento de conexão a cada requisição, ou seja, o procedimento de criptografia necessário para estabelecer a conexão segura (HTTPS) na primeira comunicação.

Com o ambiente de teste configurado, foi executado um *script* (Node.js com “axios”) que disparou **200** requisições de criação de preferência de pagamento contra a API do Mercado Pago. O processo seguiu a lógica descrita na Figura 5.

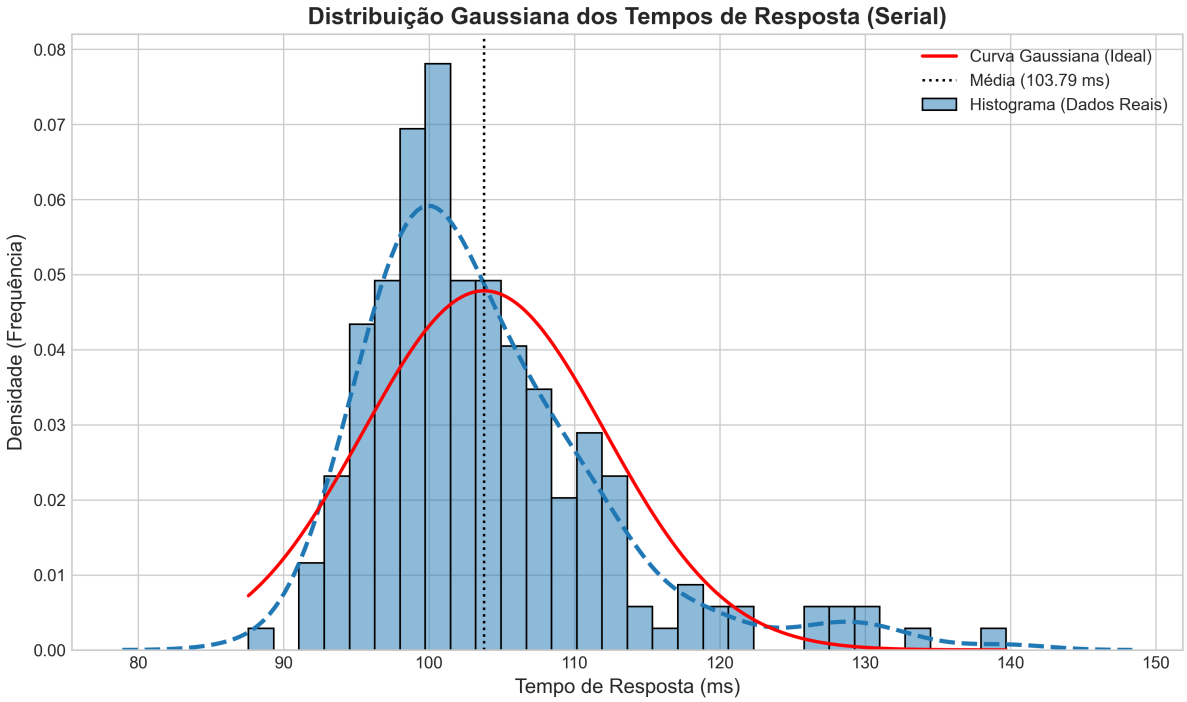
Figura 5 – Pseudocódigo do Teste de Performance Sequencial

```
// --- 1. Inicialização ---
Requer: N_Testes = 200, ACCESS_TOKEN
Inicializar temposDeResposta = [], sucessos = 0
dadosDaPreferencia = { item: 'Teste', preco: R$10.50 }

Exibir "Iniciando N testes..."

// --- 2. Execução Sequencial (em Fila) ---
PARA i = 1 ATÉ N_Testes FAÇA
    tempoInicial = tempo atual

    TENTAR
        // Envia requisição POST e aguarda
        resposta = POST("/checkout/preferences",
                        Header: "Authorization: Bearer ...",
                        Body: dadosDaPreferencia)

        // Verifica sucesso
        SE resposta == 201 ENTÃO
            sucessos = sucessos + 1
        FIM SE

    CAPTURAR erro
        Exibir "Teste #i falhou"
    FIM TENTAR

    tempoFinal = tempo atual
    duracaoMs = (tempoFinal - tempoInicial) em ms
    Adicionar duracaoMs em temposDeResposta

FIM PARA

// --- 3. Resultados ---
tempoMedio = média(temposDeResposta)
desvioPadrao = desvio padrão(temposDeResposta)
taxaSucesso = (sucessos / N_Testes) * 100

Exibir N, tempoMedio, desvioPadrao, taxaSucesso
// (Nota: Os dados brutos também são exportados para CSV)

FIM
```

Fonte: Autoria própria (2025).

A execução deste teste contra o *sandbox* do Mercado Pago gerou uma distribuição de dados que pode ser visualizada na Figura 6.

Figura 6 – Distribuição dos Tempos de Resposta (Teste Sequencial MP - 200 Requisições)



Fonte: Autoria própria (2025).

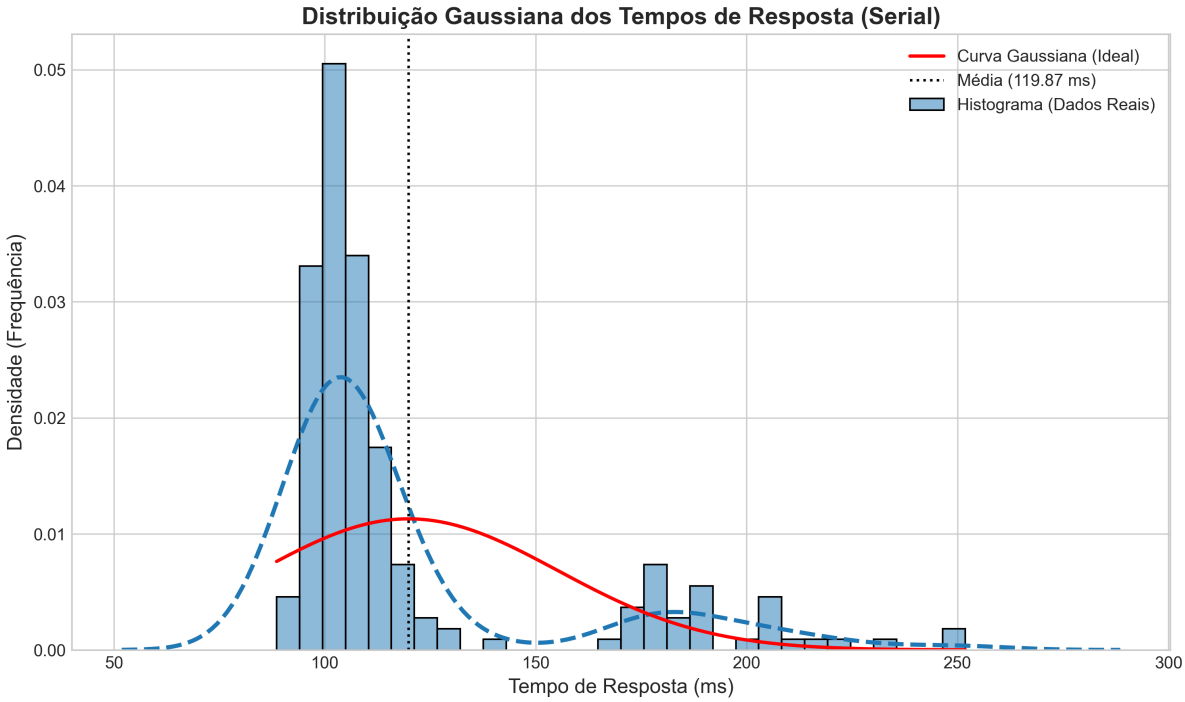
A análise do gráfico (Figura 6) confirma que a média das 200 requisições forneceu um tempo de resposta médio de 103,79 ms. É plotado um gráfico de curva gaussiana ideial para demonstrar que os dados seguem, aproximadamente, o perfil de uma distribuição normal de amostragem.

4.2.1.1 Hipótese do HTTPS afetando o tempo de resposta

A análise anterior (Figura 6), realizada com requisições imediatas, apresentou uma latência média extremamente baixa ($\sim$ 103 ms). Para testar se esse desempenho se mantém mais próximo da realidade, considerando um cenário em que o usuário navega por alguns segundos antes de tomar uma ação, foi realizado um experimento.

Neste cenário, introduziu-se um **intervalo (*delay*) de 10.000 ms (10 segundos)** entre cada uma das 200 requisições.

Figura 7 – Distribuição dos Tempos de Resposta (Teste sequencial com Intervalo de 10s)



Fonte: Autoria própria (2025)

Os resultados mostram que o perfil de resposta mudou. A distribuição dos dados deixou de ser uniforme e passou a ter um formato bimodal, com uma maior dispersão. Embora a média geral tenha sofrido um aumento discreto (para $\sim$ 119 ms), observa-se uma "cauda" de requisições superando a marca de 200 ms.

Justificativa da Variação: Os valores maiores e a instabilidade observada não indicam necessariamente uma falha no *handshake* TLS, mas são, possivelmente, resultado do atraso existente na utilização de *links* internacionais.

Considerando que a instância de teste está localizada na região `us-east-1` (EUA) e os servidores do Mercado Pago atendem principalmente o mercado latino-americano, e o tráfego passa por milhares de quilômetros de fibra óptica. É importante destacar que, diferente do primeiro teste que acontece em uma janela de tempo mais curta e captura um estado momentâneo da rede ao incluir um intervalo de 10 segundos, o experimento passa a durar mais de 30 minutos. Essa ampliação no período de amostragem aumenta bastante as chances de registrar as oscilações naturais dessas rotas internacionais e as variações momentâneas no roteamento. Por isso, é normal que os dados fiquem mais dispersos, algo que não aconteceu no teste inicial.

4.2.1.2 Recomendação Técnica

Independentemente da causa (TLS ou Latência Internacional), a existência de picos de atraso reforça a necessidade de estratégias de *frontend* resilientes.

Para um *e-commerce*, em que a percepção de velocidade é crítica, a recomendação técnica é a implementação de ***Feedback* Visual Imediato**. No momento em que o cliente clica em "Finalizar Compra", a interface deve exibir instantaneamente um indicador de progresso (*loading spinner*). Essa estratégia de UX (Experiência do Usuário) ajuda a reduzir os efeitos das oscilações na conexão e evita que o cliente desista da compra por ficar desmotivado ou inseguro, ao perceber que o processo está lento.

4.2.2 Teste 2: Execução Paralela (Mercado Pago)

O segundo teste foi um *benchmark* de carga, criado para simular um cenário de alta demanda ao *gateway* de pagamento, com o experimento de vários usuários buscando iniciar o processo de compra simultaneamente, como acontece durante um período de grande demanda, o que ocorre na *Black Friday* por exemplo.

O objetivo foi identificar limites do ambiente de *sandbox* do Mercado Pago. Para isso, outro *script* (Node.js com “axios” e “Promise.all”) foi configurado para disparar rajadas de requisições em paralelo, começando com 100 e aumentando de forma gradual até 1.000 requisições simultâneas. A lógica deste teste é descrita na Figura 8.

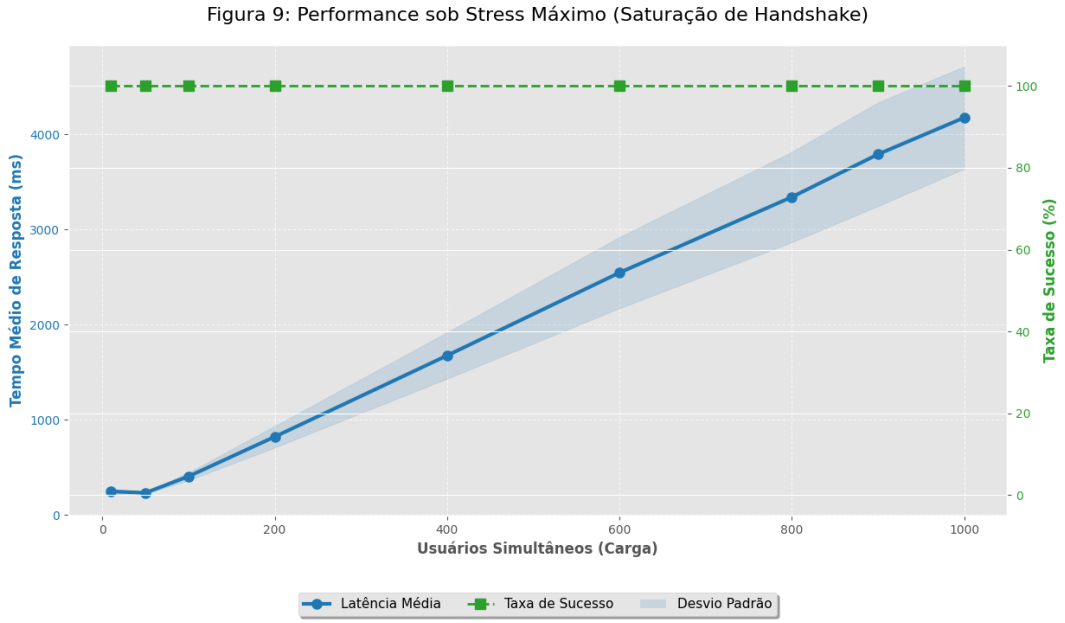
Figura 8 – Pseudocódigo do Teste de Performance Paralelo

```
// --- Função principal que dispara N testes em paralelo ---
FUNÇÃO EXECUTAR_TESTE_PARALELO(N_Testes)
    Inicializar temposDeResposta = [], sucessos = 0
    Inicializar listaDePromessas = []
    tempoGeralInicial = tempo atual

    // 1. Cria N "promessas" (tarefas) em uma lista
    PARA i = 1 ATÉ N_Testes FAÇA
        Adicionar tarefa_assincrona() à listaDePromessas
    FIM PARA

    // 2. Dispara TODAS as promessas de uma vez e espera terminarem
    ESPERAR POR TODAS(listaDePromessas)

    tempoGeralFinal = tempo atual
    duracaoTotalMs = (tempoGeralFinal - tempoGeralInicial) em ms

    // 3. Calcula estatísticas (dos resultados das promessas)
    taxaSucesso = (sucessos / N_Testes) * 100
    media = média(temposDeResposta) // Apenas dos sucessos
    desvioPadrao = desvio padrão(temposDeResposta)

    RETORNAR {N_Testes, media, desvioPadrao, taxaSucesso, duracaoTotalMs}
FIM FUNÇÃO

// --- Definição da Tarefa Assíncrona (O que cada "usuário" faz) ---
FUNÇÃO tarefa_assincrona()
    TENTAR
        resposta = POST("/checkout/preferences", ...)

        SE resposta == 201 ENTÃO
            sucessos = sucessos + 1
            Adicionar (tempo_da_resposta) em temposDeResposta
        FIM SE
    CAPTURAR erro
        // Falha registrada (sucesso não incrementado)
    FIM TENTAR
FIM FUNÇÃO
```

Fonte: Autoria própria (2025).

Os resultados da execução deste teste de carga estão plotados no gráfico da Figura 9.

Figura 9 – Desempenho da API do Mercado Pago sob Carga Paralela



Fonte: Autoria própria (2025).

A análise do gráfico de desempenho (Figura 9) mostra de forma mais clara como o sistema se comporta com o aumento da carga. Percebe-se um padrão de degradação da performance de maneira gradual e linear, mas ainda assim, o sistema continua operando de forma estável.

- **Resiliência da Taxa de Sucesso:** A linha verde demonstra que a API foi capaz de aceitar e processar **100% das requisições**, mesmo sob a carga máxima de 1.000 usuários simultâneos. Não houve recusa de conexões ou erros.
- **Crescimento Linear da Latência:** A linha azul revela uma correlação direta e proporcional entre o número de usuários e o tempo de resposta. A latência média inicia em patamares baixos (200 ms) e sobe de forma constante (linear) até ultrapassar 4.000 ms no pico de 1.000 usuários.

Este comportamento indica um gargalo de enfileiramento (provavelmente na negociação de CPU para os *handshakes* TLS). O servidor não rejeitou as requisições, mas a fila de processamento aumentou, fazendo com que cada usuário tivesse que esperar progressivamente mais tempo para ser atendido, caracterizando um cenário de saturação por capacidade de processamento, e não por política de bloqueio.

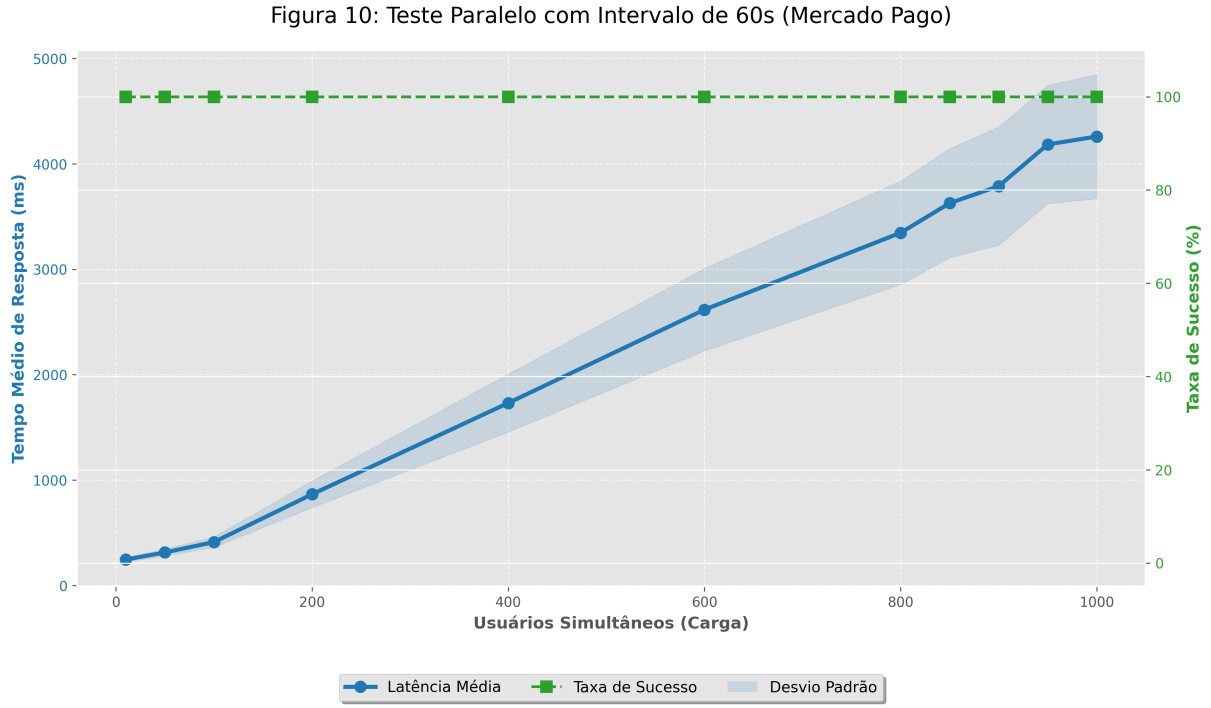
4.2.3 Segundo teste Paralelo (Mercado Pago): Verificação de Comportamento

Considerando que o teste anterior demonstrou uma resiliência total (100% de sucesso) foi observada apenas uma degradação ao longo do tempo. Por isso, investigamos se esse comportamento continua quando o tráfego acontece em rajadas espaçadas, facilitando que os mecanismos de defesa do servidor reiniciem a contagem entre os picos.

Para isso, foi executado um novo protocolo de teste introduzindo um intervalo de 60 segundos entre cada nível de carga. O objetivo desse atraso foi garantir que o servidor de *sandbox* pudesse reiniciar seus limitadores de taxa entre os testes. Além disso, a escala foi refinada para capturar com mais precisão qualquer alteração de comportamento, utilizando os níveis: [10, 50, 100, 200, 400, 600, 800, 850, 900, 950 e 1000] requisições simultâneas.

Os resultados, mostrados na Figura 10, confirmam as descobertas anteriores.

Figura 10 – Desempenho da API do MP sob Carga (Teste de Reafirmação com 60s)



Fonte: Autoria própria (2025).

A análise do gráfico de desempenho (Figura 10) demonstra a capacidade do sistema em lidar com o aumento progressivo de carga sem interromper o serviço. Observa-se um padrão de degradação linear de performance, mantendo, contudo, a estabilidade operacional com 100% de disponibilidade.

- **Resiliência da Taxa de Sucesso:** A linha verde demonstra que a API foi capaz de

aceitar e processar **100% das requisições**, mesmo sob a carga máxima de 1.000 usuários simultâneos. Não houve recusa de conexões.

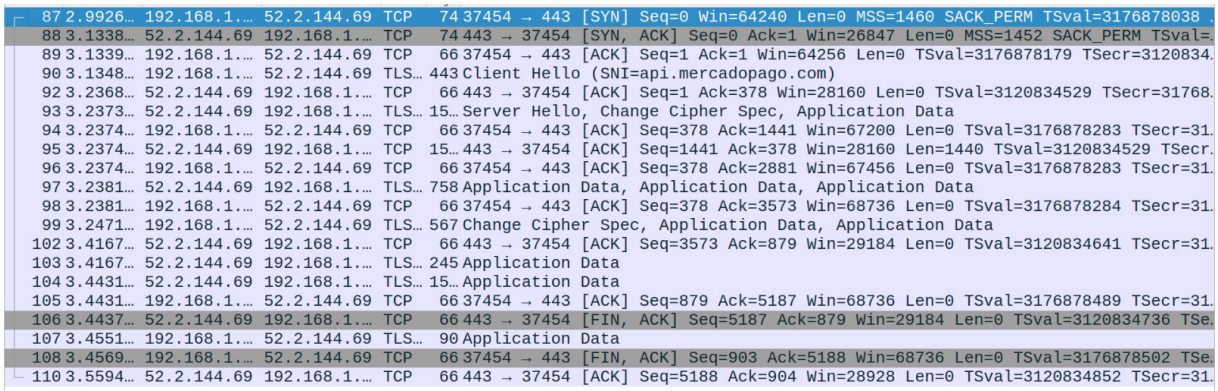
- **Crescimento Linear da Latência:** A linha azul revela uma correlação direta e proporcional entre o número de usuários e o tempo de resposta. A latência média inicia em patamares baixos (200 ms) e sobe de forma constante (linear) até atingir cerca de **4.200 ms** no pico de 1.000 usuários.

Este comportamento indica um gargalo de enfileiramento: o servidor não chegou a falhar, mas o aumento no número de requisições fez com que cada chamada precisasse esperar um pouco mais para ser processada. Isso aponta que os recursos disponíveis principalmente a capacidade de processamento estão chegando ao limite, e não que algum mecanismo de bloqueio esteja sendo acionado.

Comparando este resultado com o teste anterior, confirma-se a distinção entre a duração da conexão e a **janela de contabilização do limite de taxa**. É fundamental destacar que o intervalo de 60 segundos não influencia a latência da conexão individual. Isso se justifica pela descoberta técnica na Figura 11, que prova que a conexão TCP é efêmera e encerrada imediatamente após cada resposta.

Para validar tecnicamente o comportamento da conexão e justificar a latência observada, foi realizada a captura de tráfego de rede utilizando a ferramenta **Wireshark**. A análise dos pacotes, apresentada na Figura 11, revela o ciclo de vida exato de uma transação com a API do Mercado Pago.

Figura 11 – Captura de pacotes TCP/TLS evidenciando o fechamento da conexão



87	2.9926…	192.168.1.…	52.2.144.69	TCP	74	37454 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3176878038
88	3.1338…	52.2.144.69	192.168.1.…	TCP	74	443 → 37454 [SYN, ACK] Seq=0 Ack=1 Win=26847 Len=0 MSS=1452 SACK_PERM TSval=
89	3.1339…	192.168.1.…	52.2.144.69	TCP	66	37454 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3176878179 TSecr=3120834
90	3.1348…	192.168.1.…	52.2.144.69	TLS…	443	Client Hello (SNI=api.mercadopago.com)
92	3.2368…	52.2.144.69	192.168.1.…	TCP	66	443 → 37454 [ACK] Seq=1 Ack=378 Win=28160 Len=0 TSval=3120834529 TSecr=31768
93	3.2373…	52.2.144.69	192.168.1.…	TLS…	15…	Server Hello, Change Cipher Spec, Application Data
94	3.2374…	192.168.1.…	52.2.144.69	TCP	66	37454 → 443 [ACK] Seq=378 Ack=1441 Win=67200 Len=0 TSval=3176878283 TSecr=31
95	3.2374…	52.2.144.69	192.168.1.…	TCP	15…	443 → 37454 [ACK] Seq=1441 Ack=378 Win=28160 Len=1440 TSval=3120834529 TSecr
96	3.2374…	192.168.1.…	52.2.144.69	TCP	66	37454 → 443 [ACK] Seq=378 Ack=2881 Win=67456 Len=0 TSval=3176878283 TSecr=31
97	3.2381…	52.2.144.69	192.168.1.…	TLS…	758	Application Data, Application Data, Application Data
98	3.2381…	192.168.1.…	52.2.144.69	TCP	66	37454 → 443 [ACK] Seq=378 Ack=3573 Win=68736 Len=0 TSval=3176878284 TSecr=31
99	3.2471…	192.168.1.…	52.2.144.69	TLS…	567	Change Cipher Spec, Application Data, Application Data
102	3.4167…	52.2.144.69	192.168.1.…	TCP	66	443 → 37454 [ACK] Seq=3573 Ack=879 Win=29184 Len=0 TSval=3120834641 TSecr=31
103	3.4167…	52.2.144.69	192.168.1.…	TLS…	245	Application Data
104	3.4431…	52.2.144.69	192.168.1.…	TLS…	15…	Application Data
105	3.4431…	192.168.1.…	52.2.144.69	TCP	66	37454 → 443 [ACK] Seq=879 Ack=5187 Win=68736 Len=0 TSval=3176878489 TSecr=31
106	3.4437…	52.2.144.69	192.168.1.…	TCP	66	443 → 37454 [FIN, ACK] Seq=5187 Ack=879 Win=29184 Len=0 TSval=3120834736 TSe
107	3.4551…	192.168.1.…	52.2.144.69	TLS…	90	Application Data
108	3.4569…	192.168.1.…	52.2.144.69	TCP	66	37454 → 443 [FIN, ACK] Seq=903 Ack=5188 Win=68736 Len=0 TSval=3176878502 TSe
110	3.5594…	52.2.144.69	192.168.1.…	TCP	66	443 → 37454 [ACK] Seq=5188 Ack=904 Win=28928 Len=0 TSval=3120834852 TSecr=31

Fonte: Autoria própria (2025).

A captura comprova a inexistência de conexões persistentes (*Keep-Alive*) entre requisições distintas. É possível observar o ciclo completo em menos de 600 ms:

1. **Estabelecimento (Linhas 87-89):** Ocorre o *3-way handshake* do TCP (pacotes SYN,

SYN/ACK, ACK), estabelecendo a conexão física.

2. **Negociação de Segurança (Linha 90):** Imediatamente após, o cliente envia o *Client Hello* (TLS), iniciando o *handshake* criptográfico. Esta é a etapa mais custosa em termos de latência, consumindo aproximadamente **300 ms** do tempo total da transação, conforme indicado pelos timestamps entre o início da negociação (3.13s) e a troca de dados da aplicação (3.44s).
3. **Encerramento (Linhas 106-108):** Após a troca de dados da aplicação (*Application Data*), o servidor envia pacotes com as flags **FIN, ACK**, iniciando o encerramento imediato da conexão TCP.

Essa evidência gráfica refuta a hipótese de reutilização de conexão e confirma que cada requisição de pagamento incorre, obrigatoriamente, no custo de processamento de uma nova negociação TCP/TLS, justificando os tempos de resposta observados nos testes sequenciais.

Ou seja, tecnicamente, toda requisição já é isolada. Portanto, o sucesso deste teste deve-se exclusivamente ao fato de o intervalo ter sido suficiente para reiniciar a contagem de requisições nos servidores do *gateway* de pagamento. Isso transformou o que poderia ser interpretado como um cenário de restrição de acesso (Erro 429 pela API) em um cenário de gestão de fila no processamento do *gateway* (Latência Alta), em que a infraestrutura do provedor aceita a carga, mas demora progressivamente mais para processá-la.

4.3 Análise Stripe

Seguindo a mesma metodologia, os testes de desempenho, tanto em sequencial quanto em paralelo, foram realizados na API da Stripe. O objetivo é avaliar o tempo de resposta quando há carga no ambiente de *sandbox*. Diferente do Mercado Pago, o Stripe informa que em ambiente de teste o limite global é de apenas **25 requisições por segundo** para a maioria das operações (Stripe, 2025c).

4.3.1 Teste 1: Execução Sequencial (Stripe)

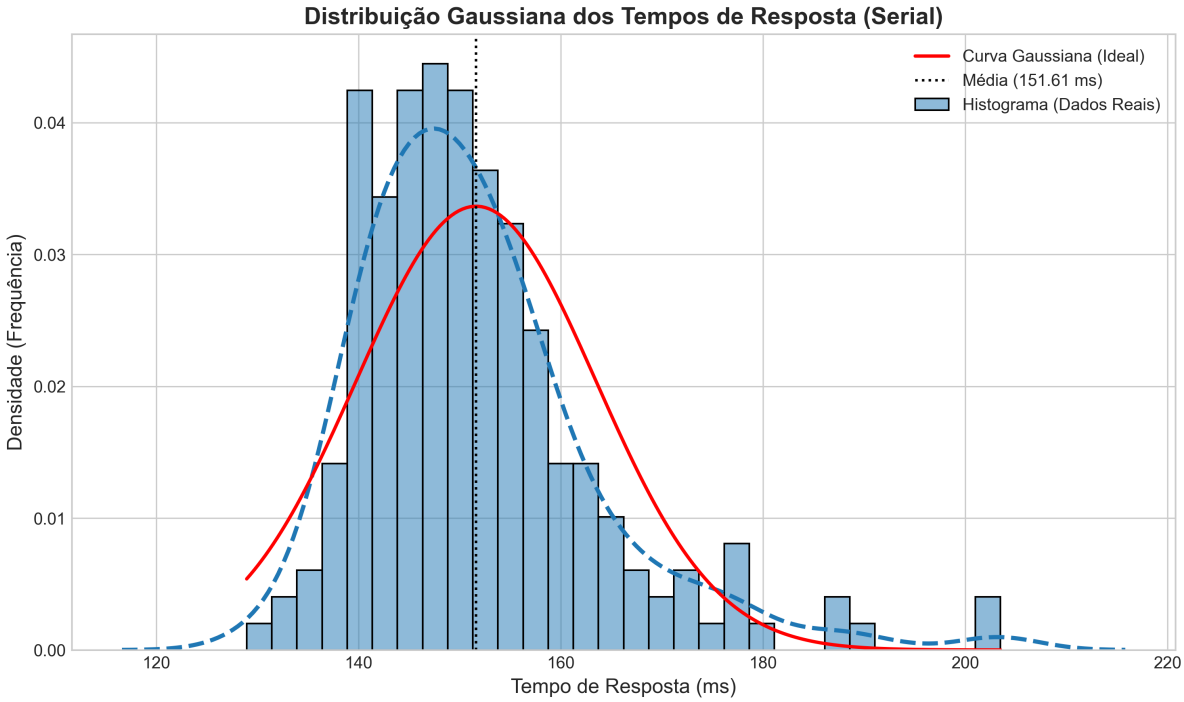
Na avaliação da Stripe, adotou-se uma metodologia que priorizou o desempenho durante o processamento contínuo. Optou-se realizar o teste de forma sequencial, sem estabelecer intervalos entre as execuções.

O objetivo foi medir o tempo de resposta padrão da API da Stripe durante um fluxo de

trabalho intenso, para entender melhor como ela se comporta em situações de uso real, em que várias requisições são feitas uma atrás da outra.

Os resultados podem ser visualizados na Figura 12.

Figura 12 – Stripe: Distribuição dos Tempos de Resposta (Teste Sequencial)



Fonte: Autoria própria (2025).

4.3.1.1 Análise do Gráfico Sequencial (Stripe)

A análise do gráfico da Stripe (Figura 12) revela uma alta consistência na gestão de novas conexões. Comparado ao Mercado Pago (Figura 6), que apresentou média de ∼100 ms, a Stripe obteve uma média ligeiramente superior, de **151,61 ms**. A distribuição apresentou-se unimodal (apenas um pico de concentração), com dados agrupados de forma estável na faixa de 140 ms a 160 ms.

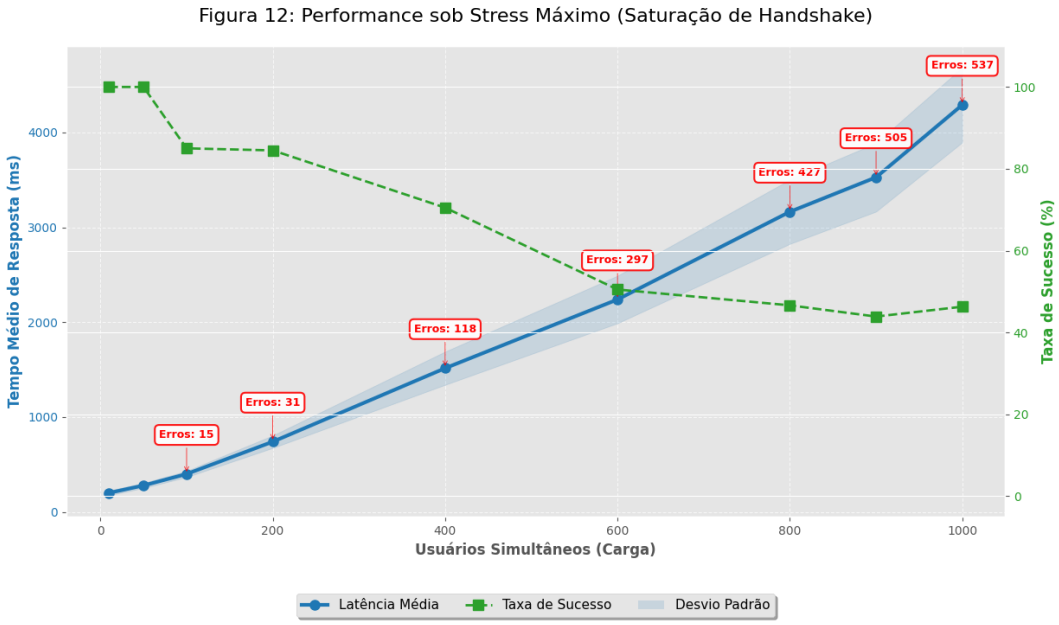
Este resultado demonstra que, embora a latência absoluta seja um pouco maior que a do concorrente neste cenário, a Stripe mantém uma previsibilidade robusta no tempo de resposta.

Apesar desses tempos serem baixos em ambiente de teste controlado, a recomendação de UX (Experiência do Usuário) permanece válida. No cenário real de produção, o tempo de resposta da API se soma à latência natural de rede entre o dispositivo do cliente e o servidor. Portanto, o uso de indicadores de carregamento (*loading spinners*) continua sendo essencial para fornecer *feedback* imediato e evitar a percepção de falta de resposta do sistema.

4.3.2 Teste 2: Execução Paralela (Stripe)

O teste em execução paralela mostrou um comportamento diferente do Mercado Pago, conforme apresentado na Figura 13.

Figura 13 – Stripe Desempenho da API do Stripe sob Carga



Fonte: Autoria própria (2025).

No gráfico de desempenho da Figura 13, revela um comportamento distinto do observado no Mercado Pago. Enquanto o concorrente priorizou o enfileiramento (aumentando a latência mas mantendo 100% de sucesso), a Stripe aplicou uma política de descarte de requisições para proteger a infraestrutura no ambiente de *sandbox*, conforme indicado na documentação.

A evolução da taxa de sucesso (linha verde) apresenta os seguintes marcos:

- **Tolerância Inicial (Até 50 requisições):** Conforme observado nos dados brutos, o sistema processou a carga de 50 requisições simultâneas com **100% de taxa de sucesso**. Isso demonstra que, apesar do limite nominal de 25 requisições por segundo citado na documentação, existe uma tolerância para picos (*bursts*) de curta duração.
- **Início do Bloqueio (100 requisições):** Ao atingir 100 requisições simultâneas, observa-se a primeira atuação do *Rate Limiter*. A taxa de sucesso sofre uma redução, fixando-se em aproximadamente **80%**, indicando que o excedente começou a receber respostas de erro (HTTP 429).
- **Saturação Progressiva (200 a 1000 requisições):** A partir de 200 usuários, a taxa de

sucesso decresce de forma constante à medida que a carga aumenta, estabilizando-se próxima a **40%** no cenário de 1.000 requisições. Diferente de um colapso total, o sistema manteve um fluxo constante de aprovações, descartando apenas o excesso que ultrapassava a capacidade de processamento do *endpoint*.

Esta execução revelou que o ambiente de teste da Stripe é resiliente, mas restritivo. A presença do cabeçalho `stripe-rate-limited-reason:endpoint-rate` nas falhas confirma que a redução na taxa de sucesso não foi um erro sistêmico, mas uma aplicação deliberada de regras de controle de fluxo.

4.3.2.0.1 Evidência Técnica: Investigação via Cabeçalhos HTTP

Para compreender a natureza exata do bloqueio, foi realizada uma inspeção nos cabeçalhos das respostas de erro (HTTP 429). A captura, apresentada na Figura 14, revelou o cabeçalho específico `stripe-rate-limited-reason`.

Figura 14 – Cabeçalho HTTP evidenciando o motivo do bloqueio (endpoint-rate)

Fonte: Autoria própria (2025).

O valor *"endpoint-rate"* confirma que o bloqueio não foi causado por um limite global da conta, mas sim por uma regra específica aplicada ao *endpoint* `/v1/payment_intents`. No entanto, é importante ressaltar que a resposta não demonstra nenhum valor numérico que quantifique esse limite (ex: requisições por segundo permitidas). Essa omissão mantém a regra de negócio obscura, demonstrando que o ambiente de *sandbox* da Stripe possui restrições granulares não apenas não documentadas, mas também não transparentes na resposta de erro, o que valida ainda mais a necessidade crítica de testes de carga práticos para dimensionar a capacidade real da integração.

Por isso, podemos perceber que, na prática, o comportamento do *Rate Limiting* durante períodos de pico costuma ultrapassar o limite de 25 requisições por segundo estipulado na documentação oficial (Stripe, 2025c). Embora os testes mostrem que há uma certa tolerância para picos curtos chegando a aceitar cerca de 40 requisições em uma única rajada simultânea,

determinar exatamente qual é esse limite não faz parte do foco deste estudo. A questão dos limites ocultos e não documentados da Stripe, que exigiriam uma engenharia reversa para serem descobertos, fica para futuras pesquisas.

4.4 Trabalhos relacionados

Enquanto o trabalho de Hisyam e Manuaba (2022) focaram na implementação funcional da integração de múltiplos *gateways* para viabilizar o *Split Payment*, este trabalho avança ao abordar os requisitos "não-funcionais". Demonstrou-se que, embora a integração funcional seja viável, a estabilidade do sistema sob carga é um fator crítico que pode inviabilizar a operação se não forem considerados os limites de *Rate Limiting* identificados nos testes.

Em uma perspectiva arquitetural, Vangala, Kasimani e Mallidi (2022) propõem modelos internos para a construção de sistemas de pagamento. Este trabalho complementa essa visão ao analisar a perspectiva do consumidor da API, o que é especialmente importante para *e-commerces* que usam serviços *SaaS (Software como Serviço)*.

Por fim, ao comparar com a análise de Nitto (2024), que avaliou *gateways* genéricos, este trabalho se diferencia ao investigar o domínio específico de *gateways* de Pagamento. Identificou-se que estes operam como “caixas-pretas” com restrições de segurança bancária e *overheads* de TLS fixos, que impõem um piso de latência independente da otimização da aplicação cliente.

Além disso, considerando o cenário do mercado brasileiro mencionado por MONTEIRO (2025), destaca que a segurança digital é um pilar fundamental para a confiança do consumidor, esta pesquisa contribui ao quantificar os custo técnico dessa segurança. Os testes sequencial demonstraram que o estabelecimento de uma conexão segura (HTTPS/TLS) impõe uma latência inicial inevitável.

No entanto, a análise de pacotes revelou que essa negociação ocorre a cada requisição (devido ao fechamento da conexão TCP). Portanto, a escolha do *gateway* não deve se basear apenas na menor taxa, mas na capacidade da infraestrutura do provedor em otimizar a negociação do protocolo de segurança, garantindo baixa latência mesmo em cenários onde novas conexões são exigidas a cada transação.

5 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho teve como objetivo fazer uma comparação entre os principais *gateways* de pagamento disponíveis no mercado brasileiro. O objetivo foi realizar uma análise dos *gateways de pagamento* com vista a explorar aspectos técnicos, fundamentais e de negócio desses sistemas.

Para isso, foi estruturado a pesquisa em duas partes: uma análise qualitativa, baseada em documentação, e uma análise quantitativa, com testes de desempenho. Assim, foi possível comparar custos e funcionalidades, mas também identificar aspectos de desempenho importantes que muitas vezes não são publicados oficialmente.

5.1 Resultados Obtidos e Resposta à Pergunta de Pesquisa

A pergunta principal desta pesquisa foi: *Quais gateways oferecem a melhor combinação de custo, funcionalidades e facilidade de integração para diferentes tipos de e-commerce.* Ao analisar os dados, é necessário realizar ponderações sobre quais aspectos são fundamentais ao negócio, como a taxa de pagamentos, tipos de pagamentos, e desempenho. Assim, cada *gateway* de pagamento precisa ser analisado com base no que o negócio de *e-commerce* necessita. Os resultados da pesquisa revelam os seguintes *trade-offs*:

- **Em Critérios de Custo (Qualitativo):** A análise documental (Tabela 1) mostrou que o **Mercado Pago** possui a taxa mais competitiva para transações PIX (0,99%), enquanto o **PagSeguro** se destaca com a menor taxa para boletos (a partir de R$ 1,89). A **Stripe**, embora com taxas de PIX mais elevadas (1,19%), demonstrou uma documentação e um ecossistema de desenvolvedor (Stripe Elements) de qualidade superior.
- **Em Performance Sequencial (Quantitativo):** Os testes seriais revelaram que a latência base do *sandbox* do Mercado Pago é ligeiramente inferior, com média de aproximadamente 103 ms em execução contínua. A Stripe, por sua vez, apresentou uma média de 151 ms no mesmo cenário. Embora ambos apresentem tempos de resposta adequados para o padrão de mercado, o Mercado Pago mostrou-se mais ágil no processamento sequencial de requisições.
- **Em Performance Paralela (Quantitativo):** Os testes de carga revelaram filosofias opostas de controle de tráfego. O *sandbox* do **Mercado Pago** priorizou a disponibilidade, suportando até 800 requisições simultâneas com 100% de sucesso. Já a **Stripe** demonstrou um *Rate Limiting* restritivo, com a taxa de sucesso reduzindo para aproximadamente **80%**

na marca de 100 requisições e atingindo cerca de **40%** no cenário de estresse máximo. Essa restrição, confirmada pela análise do cabeçalho `stripe-rate-limited-reason`, indica um ambiente projetado para validar o tratamento de erros (`HTTP 429`) e não para testes de alta volumetria.

Vale destacar que o PagSeguro não foi incluído nos testes quantitativos, pois não houve tempo suficiente e também foi difícil de configurar um ambiente de sandbox.

Com base no que foi apresentado, foram atingidos tanto o objetivo geral quanto os objetivos específicos deste trabalho. Os dados foram reunidos e organizados, os *trade-offs* envolvidos foram analisados e, ao final, foi elaborada uma recomendação técnica fundamentada nos resultados empíricos.

5.2 Recomendação Técnica

A principal orientação técnica deste trabalho não se restringe à escolha do *gateway*, mas abrange a estratégia de integração. Considerando que os experimentos realizados neste cenário específico evidenciaram uma latência inicial no estabelecimento da conexão segura (variando de aproximadamente **150 ms a 450 ms** nos dados coletados), recomenda-se a adoção de estratégias de UX (*User Experience*), como animações de carregamento (*os famosos loading spinners*) para ajudar o usuário a entender que o sistema está processando e diminuir a sensação de espera, evitando que o cliente desista do processo de compra.

5.3 Limitações do Estudo

A principal limitação desta pesquisa está no ambiente de testes. Todos os *benchmarks* de desempenho foram realizados nos ambientes de *sandbox* dos provedores. Apesar de serem ambientes controlados ideais para uma comparação metodológica, esses ambientes podem não refletir exatamente a arquitetura e a capacidade de carga dos servidores de produção, que normalmente Tem limites superiores visando alta disponibilidade do serviço. Além disso, não foi possível fazer uma análise quantitativa do PagSeguro devido às dificuldades em liberação do ambiente de testes e limitações no cronograma do estudo.

5.4 Trabalhos Futuros

Com base nas limitações que foram identificado ao longo deste estudo e nos resultados que obtidos, é possível pensar em ampliar e aprofundar essa pesquisa no futuro. Assim, novos estudos podem ajudar a entender melhor o comportamento, o desempenho e a aplicação dos *gateways* de pagamento em diferentes situações, além de fortalecer as conclusões que apresentamos aqui.

- Expandir a análise comparativa para incluir outros *gateways* importantes no mercado brasileiro, como Pagar.me, PagSeguro, AcabatePay e etc.
- Realizar mais testes de performance em ambientes de produção reais, para verificar se o *Rate Limiting* e os tempos de resposta diferem do ambiente de *sandbox*.
- Um estudo de caso mais focado na implementação da recomendação de UX *loading spinner*, utilizando testes A/B para medir de forma clara e objetiva o impacto na taxa de conversão durante o processo de *checkout*.

De forma geral, este estudo não apenas trouxe uma visão comparativa importante sobre os *gateways* de pagamento, como também revelou que os maiores desafios na integração moderna vão além da simples análise das taxas. Esses obstáculos estão mais ligados à performance percebida pelos usuários e à estabilidade técnica do sistema. A principal contribuição deste trabalho foi mostrar que o foco deve estar na experiência do usuário (UX), ajudando desenvolvedores e gestores a se prepararem melhor para melhorar as taxas de conversão de vendas. Além disso, o estudo também ofereceu uma explicação tanto conceitual quanto prática sobre como funcionam os *gateways* de pagamento, além de contextualizar os principais provedores atuantes no mercado brasileiro. Essa abordagem ajuda a entender melhor o ecossistema de pagamentos local. Espera-se que esse guia, aliado às orientações futuras, contribua para o desenvolvimento de um ecossistema de *e-commerce* no Brasil mais eficiente, resistente e voltado para o usuário.

REFERÊNCIAS

ALBERTIN, A. L. A realidade dos negócios na era digital no mercado brasileiro. 2002.

GRIGORIK, I. **High Performance Browser Networking**. [S.l.]: "O'Reilly Media, Inc.", 2013.

HISYAM, M.; MANUABA, I. B. K. Integration model of multiple payment gateways for online split payment scenario. In: IEEE. **2022 International Conference on Information Management and Technology (ICIMTech)**. [S.l.], 2022. p. 122–126.

HUSNI, E.; HIDAYAT, M. A. E-payment system using sms gateway and line application. In: IEEE. **2018 International Conference on Information and Communication Technology for the Muslim World (ICT4M)**. [S.l.], 2018. p. 173–178.

MANE, P. S. *et al.* Local service webapp with document verification. In: IEEE. **2024 Second International Conference on Advances in Information Technology (ICAIT)**. [S.l.], 2024. v. 1, p. 1–6.

MERCADO-PAGO. **Documentação para Desenvolvedores**. 2025. Disponível em: https://www.mercadopago.com.br/developers/pt/docs.

MERCADO-PAGO. **Quanto custa receber pagamentos com Checkout?** 2025. https://www.mercadopago.com.br/ajuda/33399. Acesso em: 26 set. 2025.

MONTEIRO, B. F. A evolução do comércio eletrônico no brasil: impactos, desafios e oportunidades. 2025.

NICBR. Núcleo de Informação e Coordenação do Ponto BR. **Em duas décadas, proporção de lares urbanos brasileiros com internet passou de 13% para 85%, aponta TIC Domicílios 2024**. 2024. https://nic.br/noticia/releases/em-duas-decadas-proporcao-de-lares-urbanos-brasileiros-com-internet-passou-de-13-para-85-aponta-tic-domicilios-2024. Acesso em: 22 set. 2025.

NITTO, F. N. Análise comparativa de capacidade e performance de gateways de api para gestão de relacionamento com o cliente. Universidade Estadual Paulista (Unesp), 2024.

NOTTINGHAM, M.; FIELDING, R. **Additional HTTP Status Codes**. [S.l.], 2012.

PAGBANK. **Checkout PagBank: A solução de pagamento para o seu negócio**. 2025. https://pagbank.com.br/para-seu-negocio/online/checkout. Acesso em: 26 set. 2025.

STALLINGS, W. **Cryptography and Network Security: Principles and Practice**. [S.l.]: Pearson, 2017.

STRADOLINI, C. J. Migração de sistemas monolíticos para microsserviços: estudo de caso de migração de um módulo de pagamentos de e-commerce. 2021.

STRIPE. **Pricing & fees**. 2025. https://stripe.com/br/pricing. Acesso em: 26 set. 2025.

STRIPE. **The Payment Intents API**. 2025. https://docs.stripe.com/payments/payment-intents. Acesso em: 26 out. 2025.

STRIPE, I. **Rate limits — Stripe Documentation**. 2025. Disponível em: https://docs.stripe.com/rate-limits. Acessado: 21 nov. 2025.

TYAGI, S. *et al.* Analysis and development of e-commerce web application. In: IEEE. **2022 Fifth International Conference on Computational Intelligence and Communication Technologies (CCICT)**. [S.l.], 2022. p. 65–72.

VANGALA, S. R.; KASIMANI, B.; MALLIDI, R. K. Microservices event driven and streaming architectural approach for payments and trade settlement services. In: IEEE. **2022 2nd International Conference on Intelligent Technologies (CONIT)**. [S.l.], 2022. p. 1–6.