Relatório de Arquitetura de Software: Estratégias Avançadas de Conciliação Financeira e Mapeamento de Dados na API do Mercado Pago

1. Introdução à Arquitetura de Conciliação em Ecossistemas de Pagamento

A integração de sistemas de pagamento digitais em infraestruturas corporativas de ERP (Enterprise Resource Planning) e gestão financeira constitui um dos desafios mais complexos da engenharia de software moderna. No contexto específico do ecossistema Mercado Pago, a tarefa de conciliação financeira transcende a verificação binária de status de transações; ela exige uma compreensão profunda e arquitetural da dualidade existente entre os dados *transacionais* — que representam o ato da venda e a autorização do crédito — e os dados *financeiros* — que representam a liquidação efetiva, o movimento de fundos e a consolidação bancária. Este relatório técnico, elaborado sob a perspectiva rigorosa de uma Arquitetura de Software Especializada em Meios de Pagamento, tem como objetivo dissecar exaustivamente as interfaces de programação de aplicações (APIs) do Mercado Pago para fornecer o roteiro definitivo de implementação de um sistema de conciliação robusto, auditável e escalável.

A conciliação financeira, em sua essência, é o processo de assegurar a integridade dos dados entre três domínios distintos: o registro de vendas da plataforma de origem (e-commerce ou sistema de gestão), o processador de pagamentos (Gateway/Adquirente) e o destino final dos fundos (Instituição Bancária). Em uma arquitetura de microsserviços ou sistemas distribuídos, isso se configura como um problema clássico de consistência eventual. O dado de uma venda nasce como uma "intenção" ou "autorização" em um endpoint operacional, sofre mutações devido a regras de negócio como taxas (MDR), parcelamentos e riscos de fraude, e eventualmente cristaliza-se em um lote de liquidação (settlement) que é depositado na conta bancária.

O foco central deste documento é resolver a discrepância semântica e técnica frequentemente encontrada por equipes de engenharia ao tentar reconciliar esses estados. Analisaremos a estrutura de dados JSON fornecida pelo Mercado Pago, identificaremos a taxonomia exata das taxas — incluindo MDR (Merchant Discount Rate), custos de financiamento e taxas de antecipação —, definiremos inequivocamente a "Fonte da Verdade" para a contabilidade bancária e estabeleceremos o algoritmo lógico para agrupar vendas individuais em depósitos bancários consolidados. A análise baseia-se estritamente na documentação técnica oficial e evidências empíricas de integração [1], visando mitigar riscos

de perdas financeiras decorrentes de interpretações errôneas dos campos da API.

---

2. Análise Comparativa de Endpoints: A Dualidade Transacional vs. Financeira

Uma das decisões arquiteturais mais críticas no design de um conciliador financeiro é a escolha da fonte de dados primária. A API do Mercado Pago oferece múltiplos pontos de entrada para a recuperação de informações, sendo os mais proeminentes o endpoint de busca operacional /v1/payments/search e o endpoint de relatórios financeiros /v1/settlement/report (juntamente com seus sucessores funcionais como /v1/reports/released-money). A distinção entre estes não é apenas técnica, mas conceitual, refletindo diferentes estágios do ciclo de vida do dinheiro.

2.1 O Endpoint Operacional: /v1/payments/search e a Visão Transacional

O endpoint /v1/payments/search, juntamente com a consulta direta por ID /v1/payments/{id}, atua como a interface primária para a camada transacional do sistema.[3] Sua função arquitetural é fornecer visibilidade em tempo quase real sobre o estado das transações individuais. Quando um arquiteto de software projeta a integração do checkout, é este endpoint que é consumido para responder perguntas operacionais imediatas: "O cliente pagou?", "O pedido foi aprovado?", "Houve uma recusa por fraude?".

Tecnicamente, este endpoint opera de maneira síncrona, retornando objetos JSON que representam a entidade "Pagamento" em seu estado atual. A documentação indica que ele permite buscas e retorna pagamentos realizados nos últimos doze meses a partir da data da consulta.[3] Ele oferece capacidades de filtragem por critérios como external_reference, datas de criação ou aprovação, e ordenação por diversos atributos.[3]

No entanto, para fins de conciliação bancária massiva, o uso exclusivo deste endpoint apresenta limitações severas que devem ser compreendidas:

Mutabilidade e Estado Dinâmico:
Os dados retornados por /v1/payments/search são inerentemente mutáveis. Um pagamento que hoje aparece como approved (aprovado) pode, na semana seguinte, transitar para charged_back (contestado) ou refunded (reembolsado).5 Se o sistema de conciliação basear-se apenas em uma leitura única deste endpoint no momento da venda ("snapshot"), ele perderá eventos financeiros subsequentes que alteram o saldo final. A conciliação baseada apenas neste endpoint exige um padrão de "polling" constante para verificar alterações de estado, o que é ineficiente e propenso a falhas de cobertura.
Natureza de Previsão vs. Realidade:

Embora o payload do pagamento contenha campos financeiros cruciais como net_received_amount (valor líquido recebido) e money_release_date (data de liberação do dinheiro) 4, estes valores, no momento da criação da transação, representam uma previsão do sistema baseada nas regras de liquidação vigentes. Eles não são a confirmação do depósito. Fatores externos, como bloqueios judiciais, alterações contratuais de taxas aplicadas retroativamente ou retenções manuais de risco, podem fazer com que o valor efetivamente depositado difira do valor calculado no momento da venda.
Desafios de Paginação e Consistência:
Em cenários de alto volume transacional (High Frequency Transaction), a dependência de endpoints de busca paginados (offset, limit) para extrair a totalidade das vendas diárias introduz riscos de inconsistência de dados. A latência de indexação ou "fantasmas" na paginação podem levar a transações perdidas ou duplicadas durante o processo de extração.3 O erro 1000 - Number of rows exceeded the limits é uma possibilidade real em grandes extrações 3, forçando a implementação de lógicas complexas de janelas de tempo deslizantes (range) para garantir a cobertura total.

2.2 O Endpoint Financeiro: /v1/settlement/report e a Visão de Liquidação

Em contraste com a volatilidade do endpoint de pagamentos, o endpoint /v1/settlement/report (e a família de relatórios financeiros associados, como o *Released Money Report*) representa a visão contábil definitiva e imutável.[1] Arquiteturalmente, este recurso não foi desenhado para consulta interativa, mas para a geração de artefatos de auditoria.

O funcionamento técnico deste endpoint é assíncrono. O sistema consumidor não "busca" os dados; ele "solicita" a geração de um relatório ou configura um agendamento para que o Mercado Pago gere arquivos periodicamente.[1] Estes arquivos contêm o agrupamento exato de todas as transações que compuseram um movimento financeiro específico (um lote de liquidação).

Imutabilidade e Rastreabilidade:
Uma vez que um relatório de liquidação é gerado para um período fechado, ele representa um fato contábil consumado. Ele responde à pergunta: "O que exatamente foi pago na minha conta bancária hoje?". Diferente do endpoint de busca, que lista vendas, o endpoint de relatório lista créditos e débitos realizados. Isso inclui não apenas as vendas (settlement), mas também estornos, ajustes manuais, cobranças de serviços extras e saques para conta bancária.10
Granularidade de Eventos Financeiros:
Os relatórios detalham eventos que muitas vezes são invisíveis ou difíceis de reconstruir apenas olhando para a API de pagamentos. Por exemplo, a retenção de impostos agregada ou ajustes de câmbio em vendas internacionais são explicitados nestes relatórios. O relatório de "Dinheiro Liberado" (Released Money Report), que substituiu versões anteriores como o relatório de "Dinheiro em Conta" em alguns contextos 2, é a ferramenta projetada especificamente para reconciliar o saldo disponível com os extratos bancários.

2.3 Síntese da Diferença Técnica

A diferença técnica fundamental reside no **Modelo de Consistência** e no **Propósito Arquitetural**:

- **/v1/payments/search:** É uma API RESTful de **Leitura de Estado Atual**. É otimizada para operações "quentes" (hot data), suporte ao cliente e logística. Oferece dados granulares por transação, mas sujeitos a alterações e baseados em previsões de liquidação.
- **/v1/settlement/report:** É uma API de **Geração de Artefatos Estáticos**. É otimizada para operações "frias" (cold data), contabilidade e auditoria. Oferece dados agrupados em lotes de liquidação, imutáveis e definitivos, refletindo o movimento real de fundos.

---

3. A Definição da Fonte da Verdade para Conciliação Bancária

Diante da questão **"Qual a fonte da verdade para conciliação bancária?"**, a resposta arquitetural inequívoca e mandatória para garantir a integridade financeira é: **Os Relatórios de Liquidação (/v1/settlement/report ou /v1/reports/released-money)**.

A justificativa para esta determinação baseia-se na própria definição de conciliação bancária. Conciliar banco significa explicar cada linha do extrato bancário da empresa (entradas e saídas). O extrato bancário reflete a movimentação de liquidez real. A única entidade no sistema do Mercado Pago que espelha essa movimentação de liquidez, agrupando vendas individuais em depósitos consolidados, é o Relatório de Liquidação.[1]

Utilizar a API de pagamentos (/v1/payments) como fonte da verdade para o banco é um erro metodológico conhecido como "conciliação por expectativa". Ao fazer isso, o sistema está tentando conciliar o que ele *espera* receber (baseado na venda) com o que ele *realmente* recebeu. Embora útil para identificar desvios, a fonte da verdade final para a baixa contábil deve ser sempre o documento que comprova a transferência de valores, ou seja, o relatório.

Fluxo da Verdade:

1. **Venda:** /v1/payments diz "Você vendeu X e *deve* receber Y no dia Z". (Verdade Operacional)
2. **Liquidação:** /v1/settlement/report diz "Nós transferimos Y (ou Y ajustado) para sua conta hoje, referente à venda X". (Verdade Financeira)
3. **Banco:** O extrato bancário mostra "Entrada de Y". (Verdade Bancária)

O sistema de conciliação deve ingerir os Relatórios de Liquidação para fazer o "match" com o extrato bancário, e usar o ID da transação presente nesse relatório para dar baixa nas provisões registradas via /v1/payments.

4. Engenharia de Dados do Payload: Taxas, Líquidos e Descontos

A precisão no mapeamento de valores é o coração da conciliação. Pequenas discrepâncias em arredondamentos ou na interpretação de taxas podem gerar "buracos" contábeis de grande escala ao longo do tempo. Analisaremos agora onde encontrar cada componente de custo no payload JSON do Mercado Pago.

4.1 Anatomia do Objeto fee_details e a Taxonomia de Taxas

A pergunta "Onde encontro taxa de antecipação e MDR no payload?" exige uma navegação cuidadosa pelo array fee_details presente no objeto de resposta do pagamento.[12] O Mercado Pago não apresenta um campo único "taxa total" na raiz do objeto; em vez disso, ele fornece uma lista discriminada de deduções.

O objeto fee_details é uma estrutura de array onde cada item descreve uma dedução específica aplicada ao valor bruto. Cada objeto contém tipicamente os campos type, amount e fee_payer.[12]

Mapeamento Técnico de Taxas:

Conceito de Negócio	Identificador Técnico (type em fee_details)	Descrição e Comportamento
MDR (Merchant Discount Rate)	mercadopago_fee	Esta é a taxa administrativa padrão cobrada pelo Mercado Pago pelo processamento da transação. Ela engloba o custo do adquirente, bandeira e o *spread* do gateway. É a taxa base da operação.[13]
Taxa de Antecipação	mercadopago_fee (Geralmente embutido) ou financing_fee	Em configurações padrão (D+14, D+30), o custo financeiro está embutido no MDR. Se houver uma antecipação "spot" ou

		contratada à parte que gere um custo financeiro explícito, isso pode aparecer como um incremento no mercadopago_fee ou, em casos de custos repassados, como financing_fee. A documentação de relatórios sugere que ajustes de liquidação podem ser mais detalhados nos arquivos CSV do que no JSON da API.[10]
Custo de Parcelamento	financing_fee	Se a venda foi parcelada "sem juros" para o comprador (o vendedor absorve o custo), a taxa de financiamento aparecerá aqui com fee_payer: "collector". Este valor é deduzido do líquido a receber.[10]
Comissão de Marketplace	application_fee	Se a transação ocorreu através de uma aplicação terceira que cobra uma comissão (ex: um marketplace cobrando 10% do seller), este valor aparece identificado como application_fee.[14]
Descontos/Cupons	coupon_fee	Representa a parte do valor que foi subsidiada ou descontada via cupom.[14]

Onde encontrar a Taxa de Antecipação especificamente?

É vital notar que, na API /v1/payments, o Mercado Pago tende a simplificar a visão de custos. A "Taxa de Antecipação" muitas vezes não é um campo separado se a conta está configurada para recebimento em D+0 (na hora). Nesse caso, o mercadopago_fee será simplesmente

maior do que seria em D+30. Para separar o que é "Taxa de Processamento" do que é "Taxa de Antecipação", o arquiteto deve manter uma tabela interna de taxas contratuais ou processar os Relatórios de Custos detalhados, que oferecem uma quebra analítica superior à da API transacional.

4.2 O net_received_amount e a Lógica de Descontos

A questão **"O net_received_amount considera descontos?"** pode ser respondida afirmativamente, mas com nuances cruciais sobre *quais* descontos.

O campo transaction_details.net_received_amount representa o valor final esperado para crédito na conta do vendedor *no contexto daquela transação específica*.[4] A fórmula implícita aplicada pelo Mercado Pago é:

$$NetReceivedAmount = TransactionAmount - \sum(Taxas Suportadas pelo Vendedor)$$

Análise de Cenários de Desconto:

1. **Taxas de Intermediação (mercadopago_fee, application_fee):** Sim, o net_received_amount já vem líquido dessas taxas. Se você vendeu 100 e a taxa é 5, o líquido será 95.
2. **Custos de Parcelamento (financing_fee):** Sim, se o vendedor absorve os juros ("Sem Juros" para o cliente), o net_received_amount será reduzido por esse custo.
3. **Cupons de Desconto (coupon_amount):** O tratamento é mais complexo. Se o Mercado Pago subsidia o cupom, o valor do cupom pode *entrar* como crédito. Se o vendedor dá o desconto, o transaction_amount pode já refletir o preço reduzido. O payload de exemplo [14] mostra um caso onde transaction_amount é 58, há um coupon_amount de 15, e um net_received_amount de 56. Isso indica que a lógica interna ajusta o fluxo de caixa para garantir que o vendedor receba o valor acordado, considerando quem banca o desconto.
4. **Retenções Fiscais (taxes_amount):** Em jurisdições com retenção na fonte (como IIBB na Argentina ou IVA em outros locais), o campo taxes_amount pode aparecer.[16] É crítico validar se o net_received_amount na API já deduz esses impostos. Frequentemente, retenções fiscais são aplicadas no momento do *saque* (payout) e não na transação individual, o que cria uma discrepância entre a soma dos net_received_amount das vendas e o valor efetivo do depósito. Por isso, a validação cruzada com o Relatório de Liquidação é obrigatória.

---

5. Estratégias de Agrupamento: O Link com o Extrato Bancário

A pergunta **"Existe ID único para agrupar vendas em um depósito?"** aborda um dos pontos mais dolorosos da integração. A resposta curta é: **Não existe um ID de depósito**

(Payout ID) estampado no objeto de pagamento (/v1/payments) no momento da venda.

Isso ocorre porque o depósito é um evento futuro e agregado. No momento em que a venda é criada (date_created), o Mercado Pago ainda não sabe em qual lote de transferência bancária aquela venda será incluída. Ela pode ser agrupada com vendas de amanhã, ou separada dependendo de feriados e regras de corte bancário.

5.1 O Mecanismo de Agrupamento Inverso

O "ID único" que agrupa as vendas é, na realidade, o **Identificador do Relatório de Liquidação** (ou o nome do arquivo gerado). A relação entre Venda e Depósito é de "muitos para um" e só é conhecida *a posteriori*.

O fluxo lógico para realizar esse agrupamento no sistema de conciliação é inverso:

1. **Não procure o ID do Depósito na Venda.** Você não o encontrará no endpoint /v1/payments de forma confiável para conciliação futura.
2. **Busque as Vendas no Relatório.** O processo correto é processar o arquivo gerado por /v1/settlement/report.
3. **A Chave de Link:** Dentro do relatório de liquidação, cada linha representa uma transação (crédito ou débito). O campo que liga essa linha à venda original é o source_id (que corresponde ao id numérico do pagamento) ou o external_reference (seu ID de pedido).[10]
4. **Consolidação:** Todas as linhas contidas em um único arquivo de relatório de liquidação (ou agrupadas por uma "data de liberação" comum dentro do relatório) constituem o "Lote de Depósito". A soma dos valores líquidos dessas linhas deve bater com o crédito único visto no extrato bancário.

5.2 O Algoritmo de Conciliação Recomendado

Para implementar essa lógica, o Arquiteto de Software deve desenhar o seguinte pipeline de dados:

1. **Ingestão Contínua (Operational Stream):**
   - Consumir webhooks ou realizar polling em /v1/payments/search.
   - Armazenar cada venda em uma tabela Transacoes_MP com status preliminar.
   - Dados chave: id (MP ID), external_reference (Seu ID), transaction_amount, net_received_amount (previsto), date_created, money_release_date (previsto).
2. **Ingestão de Liquidação (Financial Stream):**
   - Agendar a geração diária do relatório via API /v1/settlement/report/schedule.[9]
   - Baixar e processar o arquivo assim que disponível.
   - Para cada linha do arquivo, extrair o source_id e o valor líquido efetivamente realizado.
3. **Processo de Matching (Reconciliação):**
   - Localizar a transação na tabela Transacoes_MP usando o source_id.

- Atualizar o status para "Liquidado".
- Comparar o net_received_amount previsto (da ingestão contínua) com o valor realizado (do relatório).
- Registrar quaisquer discrepâncias (Variance Analysis) para auditoria.
- Atribuir um "ID de Lote de Conciliação" interno a todas as transações daquele arquivo.

Este método garante que o agrupamento seja exato e baseado na realidade bancária, não em suposições temporais.

---

6. Mapeamento Semântico e Dicionário de Dados

Para atender à solicitação de mapeamento exato dos campos, apresentamos a tabela abaixo. Esta referência deve ser utilizada pelos desenvolvedores para criar os objetos de transferência de dados (DTOs) no sistema de conciliação.

6.1 Tabela de Mapeamento: Conceito de Negócio vs. Campo Técnico

Conceito de Negócio (Solicitado)	**Campo Técnico na API (/v1/payments)**	**Tipo de Dado**	**Definição Técnica e Fonte**
Valor Bruto	transaction_amount	Number (Decimal)	Representa o valor total da transação cobrado do comprador. É a base de cálculo para todas as taxas subsequentes.[5]
Valor Líquido	transaction_details.net_received_amount	Number (Decimal)	O valor que entra na conta do vendedor após deduções de taxas e custos. Localizado dentro do objeto transaction_details.[3] *Nota: Sujeito a ajustes finais no*

			relatório de liquidação.
Data da Venda	date_created	String (ISO 8601)	A data e hora exata em que o pagamento foi criado/iniciado no sistema (ex: 2017-06-16T21:10:06.000-04:00).[5] Para fins de competência contábil (Accrual), usa-se esta data.
Data da Liquidação Prevista	money_release_date	String (ISO 8601)	A data calculada pelo sistema para a liberação dos fundos na conta Mercado Pago.[3] Fundamental para projeção de Fluxo de Caixa.
Taxa Total	*Não existe campo único*	Cálculo Derivado	Deve ser calculado como a soma dos valores dentro do array fee_details ou pela subtração: transaction_amount - net_received_amount.[12]

6.2 Mapeamento Auxiliar para Conciliação Avançada

Além dos campos solicitados, os seguintes atributos são indispensáveis para uma arquitetura robusta:

Campo Adicional	Campo Técnico	Utilidade na Conciliação

ID da Transação	id	Chave primária única do Mercado Pago. Usada para ligar Venda e Relatório (source_id no relatório).
Referência Externa	external_reference	O ID do pedido no seu sistema (ERP/E-commerce). Essencial para saber "o que" foi vendido.[3]
Método de Pagamento	payment_method_id	Identifica se foi visa, master, pix, account_money. Crucial para auditar se a taxa aplicada (MDR) corresponde à negociada para aquela bandeira.[10]
Tipo de Operação	operation_type	Distingue vendas online (regular_payment) de vendas físicas (pos_payment) ou assinaturas, permitindo segregar fluxos contábeis.[7]
Status Detalhado	status_detail	Fornece a razão específica do status (ex: accredited, pending_waiting_payment). Ajuda a refinar a lógica de retentativas.[7]

---

7. Conclusão e Recomendações de Implementação

A arquitetura de um sistema de conciliação financeira integrado ao Mercado Pago não deve ser subestimada como uma simples tarefa de "leitura de API". Ela exige uma estratégia de dados dual, que respeite a natureza distinta dos dados operacionais e financeiros.

Recomendações Finais do Especialista:

1. **Priorize a Imutabilidade:** Jamais altere os dados históricos de vendas baseando-se apenas em uma consulta volátil ao endpoint de busca. Use os Relatórios de Liquidação

como a "caneta final" que escreve no livro razão contábil.

2. **Automatize a Ingestão de Arquivos:** Não dependa de downloads manuais do painel. Implemente a rotina de agendamento (/v1/settlement/report/schedule) para garantir que os arquivos de conciliação sejam gerados e ingeridos automaticamente pelo sistema todas as madrugadas.[18]
3. **Auditoria de Taxas (Fee Audit):** Implemente uma "Shadow Accounting" (Contabilidade Sombra). Seu sistema deve calcular quanto *deveria* ser a taxa (baseado no contrato com o Mercado Pago) e comparar com o que vem no fee_details. Discrepâncias constantes podem indicar erros de configuração na conta do vendedor ou alterações de tabela de preços não comunicadas.
4. **Resiliência com external_reference:** Force o envio de um ID único e imutável do seu ERP no campo external_reference em todas as criações de pagamento. Em cenários de desastre (ex: perda de base de dados do conciliador), este campo é a única chave que permite reconstruir a história a partir dos dados do Mercado Pago.[3]

Ao seguir estas diretrizes, a organização estabelecerá um processo de conciliação transparente, auditável e financeiramente seguro, transformando a complexidade dos meios de pagamento em uma vantagem operacional controlada.

Works cited

1. Generate report - Account balance - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.ar/developers/en/docs/checkout-pro/additional-content/reports/account-money/generate
2. Reports - Checkout API (via Payments) - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.co/developers/en/docs/checkout-api-payments/additional-content/reports/introduction
3. Search payments - Payments - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.ar/developers/en/reference/payments/_payments_search/get
4. Get payment - Payments - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.ar/developers/en/reference/payments/_payments_id/get
5. Update payment - Payments - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.ar/developers/en/reference/payments/_payments_id/put
6. Create cancellation - Cancellations - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.ar/developers/en/reference/chargebacks/_payments_payment_id/put

7. Received payments management - Subscriptions - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.ar/developers/en/docs/subscriptions/additional-content/payment-management
8. accessed January 9, 2026, https://www.mercadopago.com.br/developers/en/docs/vtex/additional-content/reports/account-money/generate.md
9. Generate report via API - Account balance - Mercado Pago, accessed January 9, 2026, https://www.mercadopago.com.ar/developers/en/docs/checkout-pro/additional-content/reports/account-money/api
10. Overview - Checkout API (via Orders) - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.ar/developers/en/docs/checkout-api/additional-content/reports/account-money/report-fields
11. Generation by API - Released money - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.ar/developers/en/docs/reports/released-money/api
12. Update payment - Payments - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.br/developers/en/reference/payments/_payments_id/put
13. api mercadopago - GitHub Gist, accessed January 9, 2026, https://gist.github.com/aliensanderdiaz/9a717b963826315427a03f594dbe4e8e
14. Create payment - Payments - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.ar/developers/en/reference/payments/_payments/post
15. Create payment - Payments - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.br/developers/en/reference/payments/_payments/post
16. Yuju - API documentation, accessed January 9, 2026, http://docs.api.yuju.io.s3-website-us-west-2.amazonaws.com/
17. File: README – Documentation for mercadopago (2.2.0) - RubyDoc.info, accessed January 9, 2026, https://www.rubydoc.info/gems/mercadopago/2.2.0
18. Generation by API - Account balance - Mercado Pago Developers, accessed January 9, 2026, https://www.mercadopago.com.br/developers/en/docs/reports/account-money/api
19. Pedidos y opiniones - Developers Mercado Libre, accessed January 9, 2026, https://developers.mercadolibre.com.pa/en_us/order-management