**Documentação de Abertura do Projeto**

**Cash4Help**  
**Projeto – AcabouOMony – ElasticCode/Solutis**

**Enzo Jorge Soares Nascimento**   
**Matheus Yukio Makiyama**  
**Raul Gomes Reis**  
**Tales Tomaz Ito**

**São Paulo**  
**2025**

**Contexto**

Nos últimos anos, o comércio digital passou por uma transformação profunda. A facilidade de acesso a plataformas online, redes sociais e soluções de logística abriu espaço para que milhares de pequenos e médios empreendedores pudessem vender seus produtos diretamente ao consumidor final, sem depender de grandes marketplaces. Em paralelo, novas formas de interação comercial surgiram, como o Live Commerce, que combina entretenimento, engajamento em tempo real e conversão imediata. Esse modelo, que já movimenta centenas de bilhões de dólares na Ásia, está em plena expansão no Brasil, impulsionado principalmente por influenciadores digitais e vendedores autônomos que utilizam transmissões ao vivo como canal principal de vendas.

Apesar desse avanço, a infraestrutura de pagamentos ainda é um gargalo. Dados da ABComm mostram que cerca de 30% dos carrinhos de compra são abandonados na etapa de pagamento, principalmente por problemas como lentidão, exigência de múltiplos cliques, falhas na autenticação e instabilidade nas integrações. Para quem vende em tempo real, como durante uma live com centenas de espectadores simultâneos, esse tipo de fricção é fatal. A janela de conversão é curta e muitas vezes de poucos segundos e qualquer atraso ou erro técnico faz com que o cliente desista da compra e não volte mais. Além disso, muitas das soluções de pagamento existentes foram construídas para o e-commerce tradicional, com foco em experiências assíncronas e fluxos mais longos. Isso deixa uma enorme lacuna para experiências de compra rápidas, fluidas e contextuais, como as que acontecem no Live Commerce ou nas redes sociais.

Foi nesse contexto que nasceu a **Acabou o Mony**, uma fintech criada com a missão de simplificar e modernizar a forma como pagamentos são processados em ambientes digitais. O projeto parte da premissa de que a experiência de pagamento precisa acompanhar o ritmo do novo comportamento do consumidor, um consumidor que está sempre conectado, toma decisões de compra por impulso e espera agilidade, segurança e clareza em cada interação. A plataforma está sendo desenvolvida para atender a esse cenário de alta exigência, com capacidade de processar transações em tempo real, garantir segurança ponta a ponta e oferecer integrações fáceis com plataformas de Live Commerce, redes sociais, lojas virtuais e sistemas de venda direta.

O público-alvo da Acabou o Mony é composto por pequenos e médios empreendedores digitais, influenciadores que monetizam seus conteúdos por meio de vendas, plataformas de social commerce e e-commerces que buscam modernizar suas operações. Trata-se de um grupo cada vez mais relevante, tanto em volume de vendas quanto em impacto cultural, que demanda soluções tecnológicas flexíveis, acessíveis e preparadas para escalar. A proposta da plataforma não é apenas resolver um problema técnico, mas possibilitar um novo tipo de relação com o dinheiro: mais fluida, mais humana e mais conectada ao tempo real da internet.

**Objetivos**

Desenvolver um sistema que:

* Gerencie o cadastro e a conta dos usuários que irão utilizar o sistema do Acabou o Mony.
* Gerencie os cartões dos usuários cadastrados
* Administre os produtos disponíveis no e-commerce
* Registre os pedidos dos usuários de forma organizada
* Acompanhe as transações do cliente
* Mantenha a rastreabilidade das ações através de históricos
* Ajude o usuário com o proposito do projeto

**Justificativa**

Pagamentos lentos derrubam até 30% das vendas. A Acabou o Mony nasce para transformar isso em conversão com velocidade, integração e zero fricção.

**Escopo**

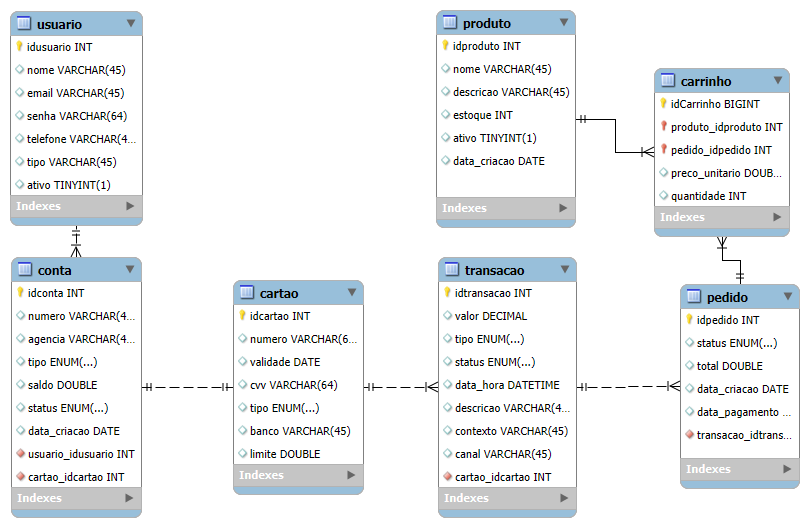
Levando em consideração todo o contexto exposto, a Cash 4 Help idealizou um projeto para otimizar os processos de pagamento em um ambiente digital. O sistema será desenvolvido com uma API em Java e contará com a estrutura de microsserviços, permitindo alta escalabilidade.

A plataforma oferecerá funcionalidades essenciais para realizar suas operações, como o cadastro e a visualização de dados pessoais, o registro e o controle das transações, além da venda de produtos através de e-commerce, com a criação de pedidos e carrinho.

Para dar início ao projeto, será necessário realizar todo o processo de documentação, detalhando aspectos fundamentais, como o contexto em que está inserido, a justificativa para sua realização, os objetivos a serem alcançados e o escopo do sistema. Essa documentação será elaborada em um documento digital no Word. Assim que todos os requisitos forem definidos no escopo, eles serão transferidos para a plataforma de gestão de projetos, o Planner, permitindo um maior controle sobre a execução das tarefas e a definição de responsabilidades dentro do grupo.

Utilizaremos a plataforma GitHub para o versionamento do código desenvolvido, garantindo um histórico detalhado das alterações e facilitando a colaboração entre os membros da equipe. Além disso, o GitHub permite o compartilhamento eficiente do código, assegurando que todos tenham acesso às versões mais atualizadas do projeto e possibilitando um fluxo de trabalho mais organizado e produtivo.

Além disso, para armazenar os dados obtidos durante o desenvolvimento do projeto, será realizada uma modelagem lógica DER das tabelas utilizadas, abrangendo desde o cadastro dos usuários utilizarão o sistema até os registros de transações e criação dos pedidos. Com base nessa modelagem, as tabelas serão criadas no banco de dados SQL Server. Para este projeto, serão utilizadas 7 tabelas com diferentes atributos, que serão apresentadas logo abaixo.

Back-End será implementado em Java com Spring Boot, contendo diversos CRUDs para atender às necessidades do sistema, através da IDE IntelliJ. Além disso, será colocado em prática a estrutura de microsserviços, além de utilizar o padrão de projeto Adapter.

Testes serão implementados durante o período de produção do sistema, desde testes unitários focados em verificar linhas de códigos individualmente, até testes de integração com o banco de dados e testes de carga, para medir e aferir o desempenho da aplicação sobre stress.

Para manter o cliente sempre informado sobre as etapas que realiza dentro da aplicação, utilizaremos um serviço de mensageria do RabbitMQ para enviar e-mails até o usuário.

**Requisitos**

**Requisitos Funcionais**

São funcionalidades que o sistema deve obrigatoriamente oferecer para cumprir seu propósito:

**RF01.** O sistema deve permitir o cadastro de usuários com informações pessoais e dados de conta.  
 **RF02.** O sistema deve permitir o gerenciamento de cartões de crédito/débito vinculados ao usuário.  
 **RF03.** O sistema deve possibilitar o registro de produtos para venda em ambiente de e-commerce.  
 **RF04.** O sistema deve permitir que usuários criem pedidos e utilizem carrinho de compras.  
 **RF05.** O sistema deve processar transações financeiras entre usuários e vendedores.  
 **RF06.** O sistema deve manter um histórico de transações para rastreabilidade.  
 **RF07.** O sistema deve exibir status das transações em tempo real.  
 **RF08.** O sistema deve enviar notificações sobre o andamento do pagamento ou falhas.

**Requisitos Não Funcionais**

Referem-se à qualidade, desempenho e restrições técnicas da plataforma.

**RNF01.** O sistema deve ser desenvolvido com arquitetura de microsserviços.  
**RNF02.** O backend deve ser implementado em Java, utilizando APIs RESTful.  
**RNF03.** O sistema deve ter tempo de resposta inferior a 2 segundos em 95% das transações.  
**RNF04.** O sistema deve garantir alta disponibilidade, mesmo em períodos de pico.  
**RNF05.** O sistema deve ser escalável horizontalmente para suportar crescimento de usuários.  
**RNF06.** A comunicação entre microsserviços deve ser segura, com uso de autenticação e criptografia.  
**RNF07.** A solução deve respeitar as diretrizes da LGPD para tratamento de dados pessoais.  
**RNF08.** O sistema deve possuir logs de auditoria para rastrear todas as operações críticas.

**Requisitos Técnicos**

**RT01.** A aplicação será dividida em serviços independentes com comunicação via HTTP ou mensageria (RabbitMQ).  
 **RT02.** O armazenamento dos dados será feito em bancos relacionais.  
 **RT03.** A autenticação será baseada em tokens (como JWT) com políticas de expiração e renovação.  
 **RT04.** Os testes de carga serão feitos através do locust.  
 **RT05.** A documentação técnica das APIs será feita via Swagger/OpenAPI.

**Premissas**

* A conexão com a internet estará disponível e estável durante o uso do sistema.
* Os dados fornecidos estarão corretos e atualizados no momento do cadastro.
* Os usuários terão acesso a um navegador compatível com o sistema (Google Chrome, Firefox ou Microsoft Edge).
* Os voluntários terão uma conta de e-mail funcional para receber notificações.
* As plataformas de Live Commerce utilizadas pelos vendedores oferecem APIs ou formas de integração com sistemas externos.
* A equipe de desenvolvimento terá acesso contínuo às ferramentas necessárias.
* A segurança de dados sensíveis, como informações de cartão e dados pessoais, será tratada conforme as diretrizes da LGPD.
* O sistema será implantado inicialmente em território nacional, considerando o comportamento e legislação brasileira.
* As tecnologias propostas (Java, microsserviços, APIs REST) são dominadas pela equipe responsável e adequadas ao escopo do projeto.

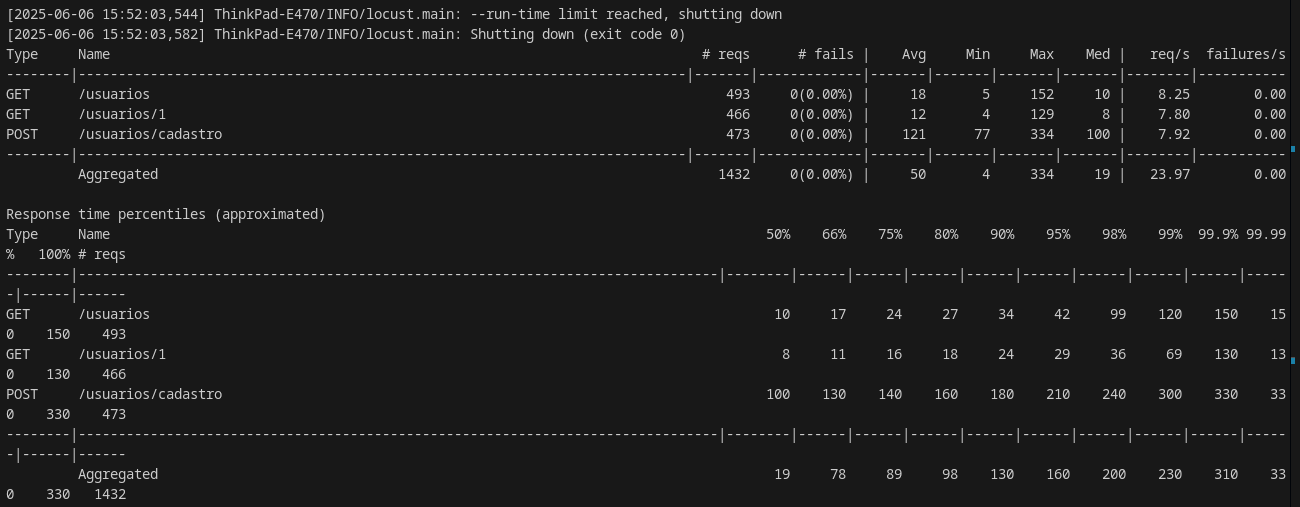
**Restrições**

* O sistema será desenvolvido apenas para ambiente web.
* O layout e a identidade visual seguirão um padrão simples, respeitando o tempo disponível.
* A segurança do sistema será garantida em nível básico, sem criptografia avançada ou certificação digital.
* O escopo de funcionalidades está restrito àquelas descritas no documento de objetivos, recursos extras não serão considerados sem redefinição formal do escopo.
* A infraestrutura de hospedagem será limitada ao ambiente disponível para o grupo, o que pode restringir testes de carga em grande escala.

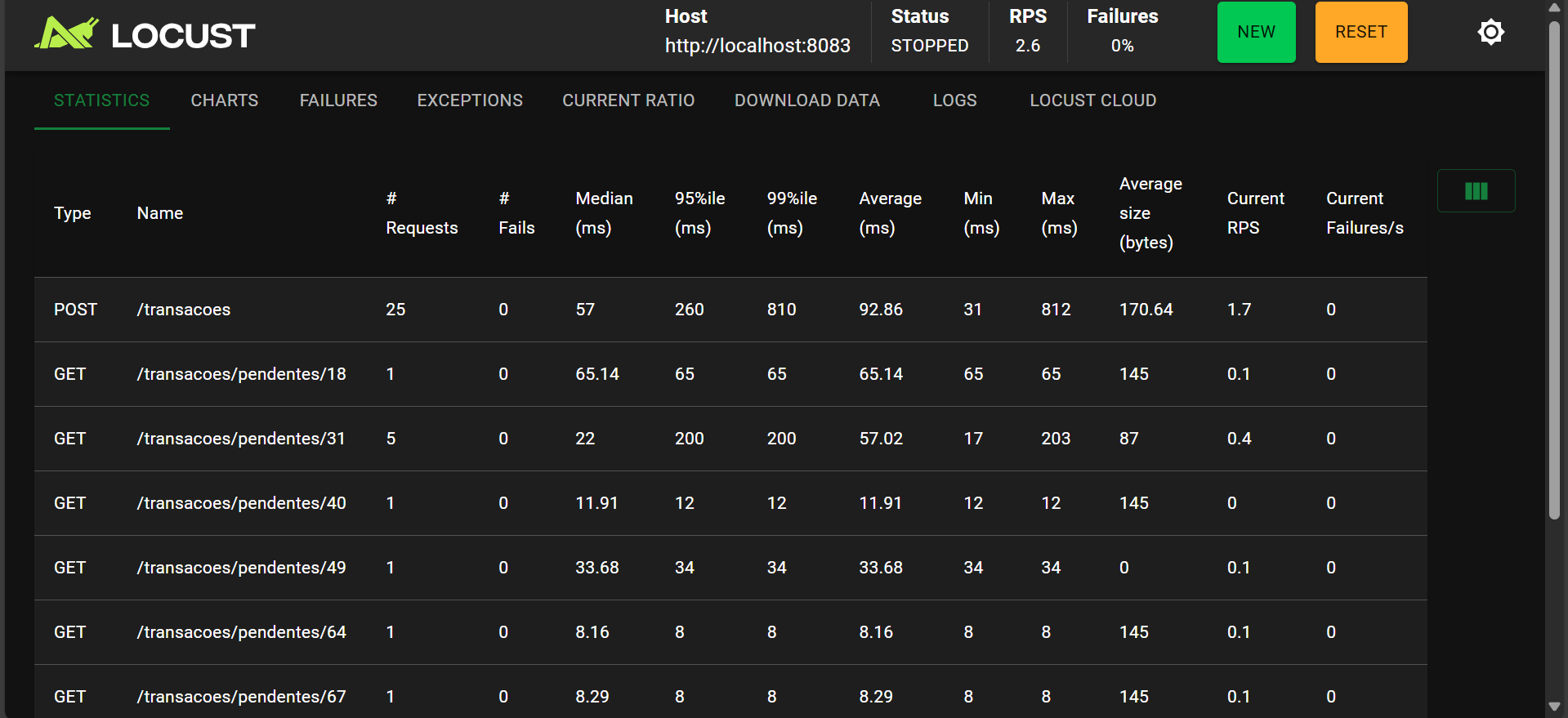
**Artefatos relacionados à regra de negócio**

**Teste de Carga**

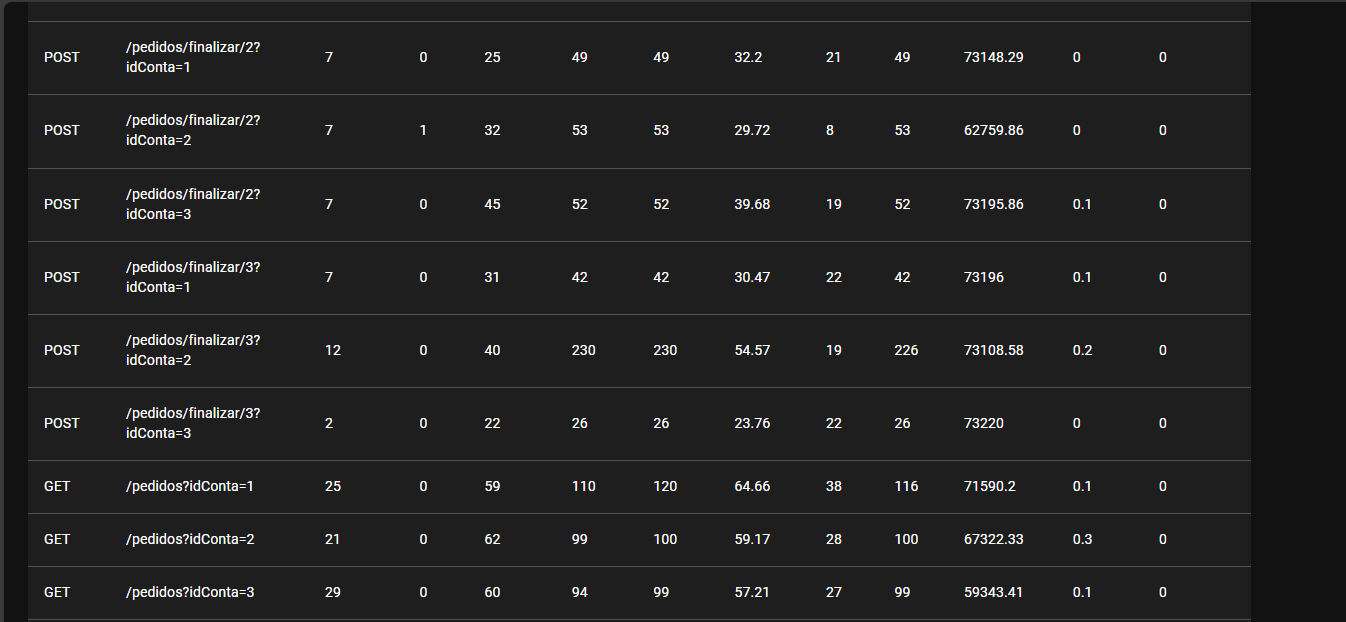
**Usuário**



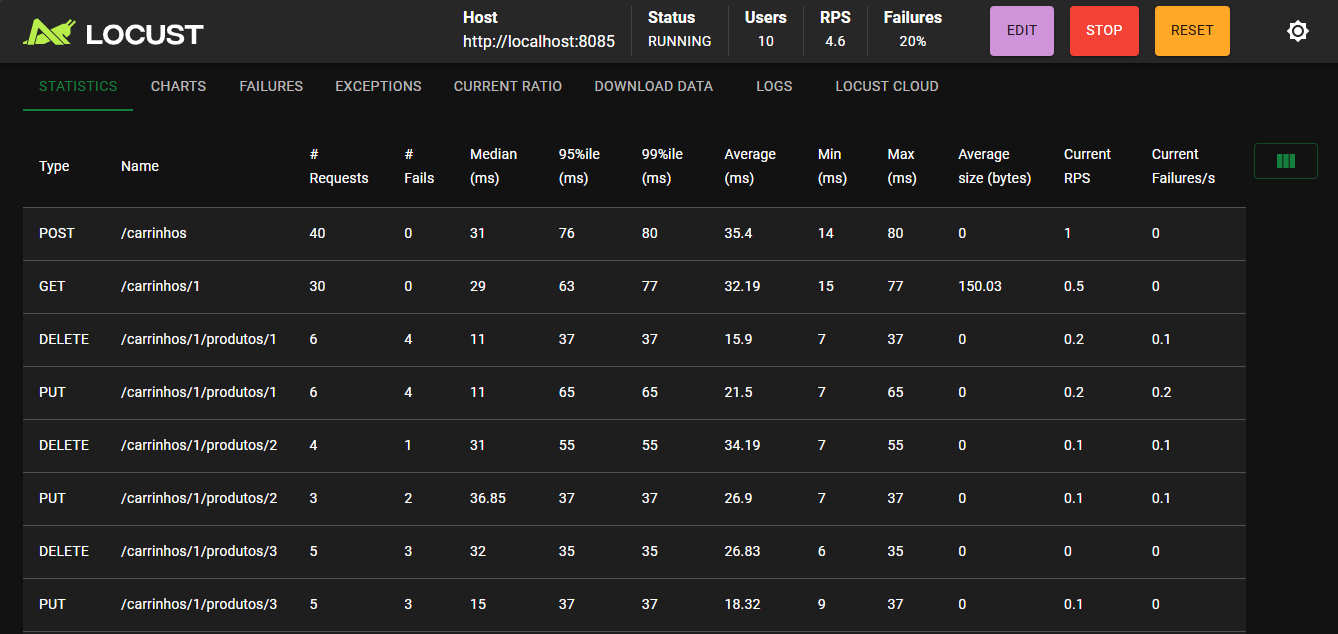
**Transação**



**Pedido**



**Carrinho**



**Produto**



**Diagrama de solução**

[Diagrama de solução acabou o mony – Figma](https://www.figma.com/design/wXU8fkqlJP2N500v0qACj6/Diagrama-de-solu%C3%A7%C3%A3o-acabou-o-mony?node-id=0-1&p=f&t=nvDFhTrXnna4Xaf0-0)

**Ferramentas de planejamento**

Para ferramentas de planejamento de projeto utilizamos:

* Microsoft Planner para separação de entregas por sprints, product backlog e controle de entregas por buckets to do, doing e done.
* Git e GitHub para organização e versionamento de código.

**Link do projeto no GitHub**

[acabou-o-mony](https://github.com/acabou-o-mony)

**Referências Bibliográficas**

**NUVEMSHOP**. Relatório de Impacto 2023. São Paulo: Nuvemshop, 2023. Disponível em: https://www.nuvemshop.com.br/blog/relatorio-impacto-2023. Acesso em: 6 jun. 2025.

**ABCOMM – Associação Brasileira de Comércio Eletrônico**. Relatório Anual de E-commerce 2023. São Paulo: ABComm, 2023. Disponível em: https://www.abcomm.org/relatorio-ecommerce-2023. Acesso em: 6 jun. 2025.

**IRESEARCH. China Live Streaming E-commerce Report 2023**. Pequim: iResearch, 2023. Disponível em: https://www.iresearchchina.com/content/details8\_12345.html. Acesso em: 6 jun. 2025.