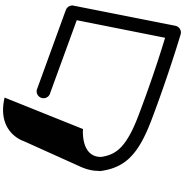


Documentação - **Paper Bank**

Equipe:
Rhuan Soares Ramos,
Thauan Fabrício da Rocha e
Gabriel de Oliveira Souza

1. Introdução

Este documento apresenta de forma completa e detalhada o desenvolvimento do sistema do banco digital *Paper*, ou seja, apresenta a análise e a modelagem do sistema. A instituição financeira foca em eficiência e flexibilidade como forma de atender ao público jovem e adulto de maneira bem estruturada. Neste sentido, como forma de descrever a construção do sistema, serão abordados ao longo dos tópicos informações fundamentais para delimitar as ideias, incluindo o minimundo, o escopo inicial, o levantamento dos atores e envolvidos no processo, além dos requisitos do sistema e o diagrama de casos de uso.

Assim, este documento tem como principal propósito servir como referência para o entendimento funcional e técnico do sistema, sendo essencial para orientar as próximas etapas de desenvolvimento, validação e implementação da solução.

2. Visão Geral do Produto

Neste tópico, será apresentada de maneira detalhada a descrição, a proposta do produto com suas delimitações, seu objetivo e seu escopo.

2.1. Minimundo

O banco *Paper* é um sistema bancário estruturado para oferecer recursos práticos, ágeis e acessíveis, atendendo plenamente as necessidades financeiras de seus usuários. O sistema atuará na gestão e processamento de transações bancárias (transferências, débito, crédito e PIX), registro de operações financeiras (extrato e comprovantes de pagamento), concessão de empréstimos, uso de crédito e cobranças futuras, fornecimento de cartões, poupança/investimentos e financiamentos.

Por outro lado, o *Paper* não será responsável por gerenciar recursos humanos, fiscalizações internas ou integrações externas complexas, permanecendo restrito às operações financeiras dos clientes, saques por meio de caixa eletrônico e depósitos por meio de boletos bancários. O sistema busca oferecer uma experiência intuitiva, segura e confiável, permitindo que os usuários realizem suas transações, acompanhem seu histórico financeiro e utilizem serviços de crédito e investimento com facilidade.

2.2. Delimitação do Escopo Inicial

O escopo inicial do sistema *Paper* abrange as funcionalidades essenciais de um banco digital, permitindo aos usuários realizar operações financeiras básicas de forma prática e segura.

As funcionalidades do usuário incluem:

- Cadastro e login em contas pessoais;
- Transações financeiras, como débito, crédito e transferências via PIX;
- Consulta de histórico e dados bancários;
- Solicitação de empréstimos, cartões, financiamentos e investimentos.
- Saques no Banco 24 Horas e depósitos no boleto bancário.

No âmbito administrativo, o sistema oferecerá:

- Gestão das informações cadastrais e financeiras dos clientes;
- Acesso a relatórios gerenciais;
- Manutenção do banco de dados;
- Todas as operações respeitam integralmente a Lei Geral de Proteção de Dados (LGPD).

O sistema não contemplará, nesta etapa:

- Integração com outros bancos ou serviços externos;
- Funcionalidades de programas de pontos, seguros, consórcios, capitalização, carteira digital ou recargas de créditos de telefonia;
- Atendimento via chat ou call center;

Essas possibilidades poderão ser consideradas em futuras expansões, mas não fazem parte da presente delimitação do escopo inicial.

3. Atores (Usuários) e Envolvidos

- **Cliente** - Utiliza o sistema para realizar suas ações financeiras, como criar uma conta, consultar saldo, fazer transferências, depositar dinheiro e solicitar empréstimos. É o principal usuário do sistema.
- **Gerente** - Utiliza o sistema para acessar informações privilegiadas, como relatórios gerenciais, a fim de tomar decisões estratégicas. Também concede ou nega permissões às solicitações dos usuários, supervisiona operações bancárias, mantém e realiza a manutenção dos dados, assegurando o bom funcionamento do sistema. Além disso, acessa repositórios para obter relatórios estatísticos das ações dos usuários, identificando lacunas e promovendo ajustes na plataforma.
- **Agente de suporte** - Interage com o sistema para ajudar o cliente na resolução de problemas, como dificuldades no acesso à conta ou dúvidas sobre transações e funcionalidades.

- **Banco central** - Instituição reguladora que não interage diretamente com o sistema, mas impõe as regras de negócio e a legislação (como a LGPD) que o sistema do banco Paper deve seguir.

4. Requisitos do Sistema

4.1. Regras de Negócio

Na atual sessão, serão classificadas por número, nome e descrição cada uma das regras de negócio direcionadas à proposta de sistema do banco *Paper*, promovendo uma visão ampliada do mecanismo interno do software baseado nas demandas do cliente, tendo em vista as delimitações do escopo inicial.

N°	Nome	Descrição
RN01	Conta unitária	Cada CPF estará atrelado à apenas um cadastro de usuário.
RN02	Senha de acesso	Cada usuário deverá criar uma senha numérica de 6 dígitos para acesso à conta.
RN03	Desbloqueio de conta	O desbloqueio da conta somente poderá ser realizado mediante resposta de pergunta de segurança.
RN04	Horário para transferências de alto valor	Transferências acima de limite definido pelo banco só poderão ser realizadas em horário comercial.
RN05	Erros em transferências	Erros em transferências poderão ser revertidos mediante contato com o gerente, respeitando prazos internos.
RN06	Reembolso	Solicitações de reembolso serão avaliadas e tratadas em prazo de 48hrs no horário comercial.
RN07	Saques por meio de caixa eletrônico	Saques poderão ser realizados em qualquer horário.
RN08	Depósitos por meio de boleto bancário	A emissão de boletos para depósitos poderão ser realizados em qualquer horário, caso esteja fora do horário comercial e o boleto seja pago, o pagamento será automaticamente agendado para um horário comercial.
RN09	Concessão de empréstimo	Empréstimos só serão concedidos se o cliente não ultrapassar percentual de endividamento definido pelo banco.

RN10	Conta com saldo negativo	O sistema permitirá saldo negativo, limitado às condições de crédito estabelecidas.
RN11	Concessão de financiamento	Financiamentos só serão concedidos se a dívida total do cliente estiver dentro do percentual permitido.
RN12	Dados obrigatórios para financiamento	Financiamentos exigirão a apresentação de dados completos de identificação do cliente.
RN13	Prazo de entrega de cartões	Cartões solicitados terão prazo de entrega de no máximo 1 semana.
RN14	Valores de investimento	Poderão ser realizados investimentos a partir de R$ 0,01.
RN15	Investimentos com saldo negativo	Não será permitido realizar investimentos quando a conta estiver negativa.
RN16	Imposto de Renda sobre investimentos	Todo investimento com rendimento estará sujeito à cobrança de IR.
RN17	IOF sobre investimento	O IOF será aplicado de forma regressiva conforme o tempo de aplicação.
RN18	Rendimentos de investimentos	Todos os investimentos renderão juros proporcionais ao CDI vigente.
RN19	Contato com gerente	O cliente poderá contatar seu gerente diretamente pelo site ou telefone informado no cadastro.
RN20	Relacionamento usuário - gerente	Cada cliente terá um gerente exclusivo; cada gerente poderá atender múltiplos clientes.
RN21	Prazo de atendimento gerencial	Solicitações dos clientes ao gerente deverão ser atendidas em prazo definido pelo banco.
RN22	Atendimento gerencial por urgência	Demandas simples ou médias serão atendidas em horário comercial; demandas graves a qualquer momento.
RN23	Proteção de dados	Todos os dados armazenados estão sujeitos à LGPD, sem exceções.
RN24	Retenção de histórico após encerramento	Após encerramento da conta, o histórico será mantido pelo prazo legal exigido.

4.2. Requisitos Funcionais

N°	Nome	Descrição
RF01	Bloqueio por tentativas inválidas	Após 3 tentativas incorretas de login, a conta será temporariamente bloqueada por segurança.
RF02	Recuperação de Acesso	O cliente poderá recuperar o acesso à conta através de validação via e-mail ou SMS cadastrado.
RF03	Dados obrigatórios para cadastro	O cadastro exigirá: e-mail, telefone, CPF, nome completo e CEP do usuário.
RF04	Dados bancários vinculados	Todo cadastro será associado a um número de agência e conta.
RF05	Validação de primeiro acesso	No primeiro acesso, será enviado um código de segurança ao e-mail cadastrado.
RF06	Redefinição de senha	A redefinição de senha deverá ser realizada por link enviado ao cliente.
RF07	Autenticação em transferências	Para toda transferência, deverá ser fornecida a senha de 6 dígitos cadastrada.
RF08	Autenticação em dois fatores (2FA)	O sistema poderá exigir autenticação em dois fatores para transações de alto valor.
RF09	Transferências incompletas	Transferências iniciadas e não concluídas serão automaticamente canceladas em até 10 minutos.
RF10	Cadastro de chave PIX	O cadastro de chave PIX exigirá um identificador válido do usuário.
RF11	Transferências via PIX	Para transferências via PIX, será obrigatória a indicação da chave do destinatário.
RF12	Transferências programadas	O sistema permitirá agendamento de transferências, respeitando prazo máximo definido pelo banco.
RF13	Operações suspeitas	Toda operação considerada suspeita gerará notificação imediata em todos os canais cadastrados do cliente.
RF14	Identificação em saques e depósitos	Para saques e depósitos será obrigatória a apresentação de dados pessoais do cliente.
RF15	Consulta de transações	O usuário poderá consultar seu histórico de transações a qualquer momento.
RF16	Consulta de dados pessoais e bancários	O usuário poderá consultar seus dados pessoais e bancários cadastrados a qualquer momento.
RF17	Gerenciamento de limites	O sistema permitirá ao cliente consultar e ajustar limites de transferências e saques.

RF18	Consulta de parcelamento	Valores parcelados serão atualizados em tempo real e disponibilizados para consulta.
RF19	Pagamentos no débito de alto valor	Pagamentos no débito acima de limite definido serão confirmados via e-mail ou SMS.
RF20	Solicitação de empréstimo	Empréstimos deverão ser solicitados pelo sistema e avaliados dentro de prazo definido.
RF21	Solicitação de cartão	Cartões poderão ser solicitados via sistema e serão processados apenas em horário comercial.
RF22	Bloqueio e desbloqueio de conta/cartão	O cliente poderá solicitar bloqueio ou desbloqueio de conta ou cartão em caso de perda, roubo ou fraude.
RF23	Resgate de investimentos	Valores investidos poderão ser resgatados a qualquer momento, em horário comercial.
RF24	Gestão administrativa	A equipe de gestão poderá acessar e monitorar dados cadastrais e transações a qualquer momento.
RF25	Armazenamento de relatórios	Todos os relatórios de transações serão enviados ao e-mail do gerente responsável.
RF26	Manutenção de dados	A equipe técnica poderá realizar manutenção no banco de dados conforme permissões internas.
RF27	Encerramento de Conta	O cliente poderá solicitar encerramento de conta, desde que não haja pendências financeiras.
RF28	Reversão de transferência	O sistema deve permitir que um Gerente, com a devida autenticação, reverta uma transferência financeira.

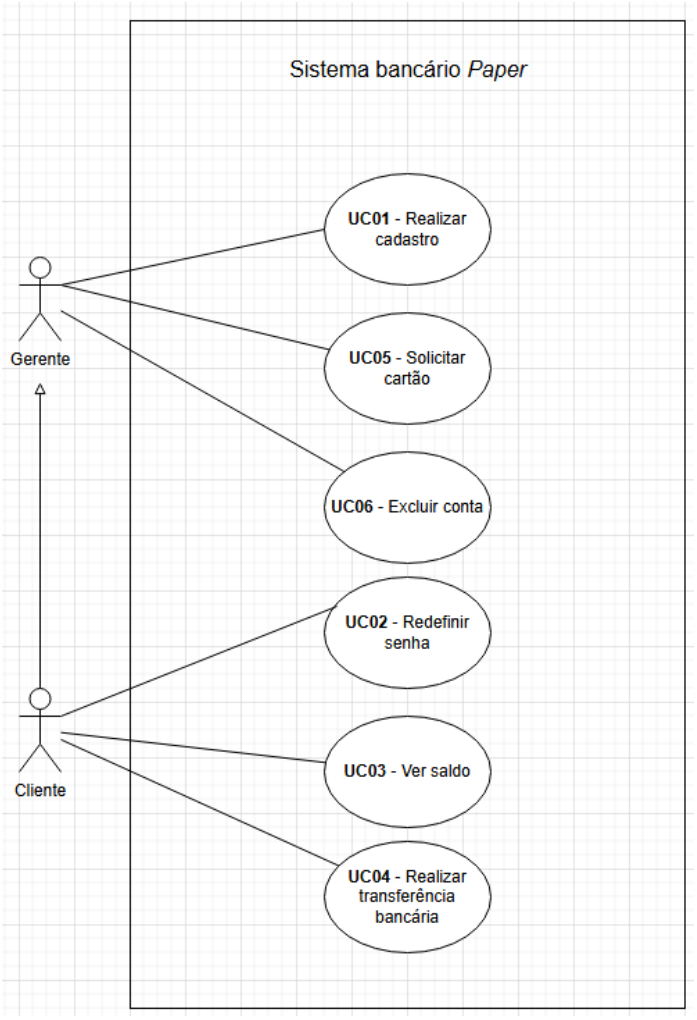
4.3. Requisitos Não-Funcionais

N°	**Nome**	**Descrição**
RNF01	Encerramento de sessão por inatividade	Sessões sem atividade por mais de 10 minutos serão encerradas automaticamente.
RNF02	Interface Intuitiva	Um usuário que já utiliza internet banking deve ser capaz de realizar uma transferência PIX em até 5 cliques, sem precisar de um manual. A interface deve seguir princípios básicos de design (ex: botões claros, feedback visual).
RNF03	Responsividade	A interface web deve se adaptar para ser

		utilizável em desktop, tablet e smartphone, garantindo que a demonstração funcione em qualquer dispositivo.
RNF04	Auditoria e Rastreabilidade	Todas as transações financeiras (login, transferência, saque) devem ser registradas em um log com data, hora, usuário e valor, para permitir a auditoria em caso de inconsistências.

5. Casos de Uso

5.1. Diagrama de Caso de Uso



5.2. Descrição dos Casos de Uso

UC01 - Realizar cadastro	
Descrição/Objetivo	Permitir que um novo usuário crie uma conta bancária no sistema Paper, fornecendo os dados necessários e estabelecendo as credenciais de acesso.
Tipo de UC	Primário
Atores	Cliente(Primário), gerente(Secundário).
Pré-Condições	● O usuário não pode possuir uma conta existente vinculada ao seu CPF. ● O usuário deve ter acesso à internet e possuir um e-mail e telefone válidos.
Pós-Condições:	● A conta do cliente é criada e ativada no sistema. ● Um número de agência e conta é gerado e vinculado ao cadastro. ● O cliente é associado a um gerente exclusivo. ● O histórico do cliente começa a ser registrado.
Trigger:	
Fluxo Básico:	1. O cliente inicia o processo de cadastro. 2. O sistema solicita os dados obrigatórios: e-mail, telefone, CPF, nome completo e CEP. 3. O sistema valida que o CPF informado ainda não está cadastrado. 4. O sistema solicita a criação de uma senha numérica de 6 dígitos. 5. O sistema envia um código de segurança ao e-mail cadastrado para validação do primeiro acesso (RF05). 6. O cliente insere o código de segurança recebido. 7. O sistema valida o código, cria a conta, associa os dados bancários a um gerente. 8. O sistema exibe uma mensagem de boas-vindas e confirmação do cadastro.
Fluxos Alternativos:	● **FA01 - CEP inválido (inicia no Passo 2)**: Se o CEP for inválido, o sistema solicita que o cliente insira o endereço manualmente.
Fluxos de Exceções:	● **FE01 - CPF já cadastrado (inicia no Passo 3)**: Se o sistema identificar que o CPF já existe, o fluxo é interrompido e uma mensagem informa que já existe uma conta para este documento. ● **FE02 - Código de validação incorreto (inicia no Passo 6)**: Se o cliente inserir um código de validação incorreto, o sistema informa o erro e permite uma nova tentativa ou o reenvio do código.
Referências Cruzadas (Requisitos Associados):	RN01, RN02, RN20, RF03, RF04, RF05.

UC02 - Redefinir senha	
Descrição/Objetivo	Cliente deseja trocar a antiga senha por uma nova senha.
Tipo de UC	Primário
Atores	Cliente(Primário).
Pré-Condições	● O cliente deve possuir uma conta bancária ativa. ● O cliente deve ter acesso ao e-mail ou SMS cadastrado na conta.
Pós-Condições:	● A senha antiga do cliente é invalidada. ● A nova senha definida pelo cliente passa a ser a credencial de acesso válida.
Trigger:	
Fluxo Básico:	● O cliente seleciona a opção para redefinir a senha. ● O sistema solicita um dado de identificação (ex: CPF). ● Após a identificação, o sistema envia um link de redefinição para o e-mail ou SMS cadastrado do cliente (RF06, RF02). ● O cliente acessa o link recebido. ● O sistema direciona o cliente para uma página segura onde ele deve digitar e confirmar a nova senha numérica de 6 dígitos (RN02). ● O sistema valida a nova senha e a atualiza no cadastro do cliente. ● O sistema exibe uma mensagem de sucesso, informando que a senha foi alterada.
Fluxos Alternativos:	N/A
Fluxos de Exceções:	● **FE01 - Dados de identificação não encontrados (inicia no Passo 2)**: Se o CPF informado não corresponder a nenhuma conta, o sistema exibe uma mensagem de erro. ● **FE02 - Link de redefinição expirado (inicia no Passo 4)**: Se o cliente tentar usar um link que já expirou, o sistema informa o ocorrido e orienta a iniciar o processo novamente.
Referências Cruzadas (Requisitos Associados):	RN02, RF02, RF06.

UC03 - Ver saldo	
Descrição/Objetivo	Acessar o saldo da conta bancária.

Tipo de UC	Primário
Atores	Cliente(Primário).
Pré-Condições	● O cliente deve estar autenticado (logado) no sistema.
Pós-Condições:	● Nenhuma alteração de estado no sistema. O saldo é apenas exibido para o cliente.
Trigger:	
Fluxo Básico:	1. O cliente acessa a área principal da sua conta após o login. 2. O sistema automaticamente busca e exibe o saldo atual da conta do cliente em um local de destaque na interface.
Fluxos Alternativos:	N/A
Fluxos de Exceções:	**FE01 - Falha na comunicação com o servidor (inicia no Passo 2)** : Se houver um problema de conexão, o sistema exibe uma mensagem informando que não foi possível carregar o saldo e sugere tentar novamente.
Referências Cruzadas (Requisitos Associados):	RF16.

UC04 - Realizar transferência bancária	
Descrição/Objetivo	Cliente deseja transferir dinheiro da sua conta para outra conta.
Tipo de UC	Primário
Atores	Cliente(Primário).
Pré-Condições	● O cliente deve estar logado no sistema. ● O cliente deve ter saldo suficiente para a transferência ou estar dentro do seu limite de crédito (RN10). ● A conta não pode estar bloqueada.
Pós-Condições:	● O valor da transferência é debitado da conta do cliente. ● O valor é creditado na conta do destinatário. ● A transação é registrada no histórico (extrato) de transações do cliente.
Trigger:	
Fluxo Básico:	1. O cliente escolhe o tipo de transferência (ex: PIX). 2. O sistema solicita a chave PIX do destinatário (RF11).

	3. O sistema busca e exibe os dados do titular da chave para confirmação pelo cliente. 4. O cliente confirma os dados e informa o valor a ser transferido. 5. O sistema verifica se o valor e o horário da transação estão de acordo com as regras de segurança (RN04, RF17). 6. O sistema solicita a senha de 6 dígitos do cliente para autorizar a transação (RF07). 7. O cliente insere sua senha. 8. O sistema valida a senha, processa a transação, debita o valor da conta de origem e credita na de destino. 9. O sistema exibe o comprovante da transação e o disponibiliza para consulta (RF15).
Fluxos Alternativos:	● **FA01 - Agendar transferência (inicia no Passo 4)** : O cliente pode escolher uma data futura para a transação, e o sistema a agendará (RF12). ● **FA02 - Transferência de alto valor (inicia no Passo 5)**: Para transferências de alto valor, o sistema pode solicitar um segundo fator de autenticação (2FA) (RF08).
Fluxos de Exceções:	● **FE01 - Saldo insuficiente (inicia no Passo 5)** : Se o saldo for insuficiente (e não houver limite), o sistema bloqueia a operação e informa o cliente. ● **FE02 - Chave PIX inválida/não encontrada (inicia no Passo 2)**: O sistema informa que a chave é inválida e solicita a correção. ● **FE03 - Senha de transação incorreta (inicia no Passo 6)** : O sistema informa o erro. Após 3 tentativas, a conta pode ser bloqueada (RF01). ● **FE04 - Transferência não concluída (inicia após Passo 7)**: Se a transação não for concluída em 10 minutos, será cancelada (RF09).
Referências Cruzadas (Requisitos Associados):	RN04, RN10, RF01, RF07, RF08, RF09, RF11, RF12, RF15, RF17.

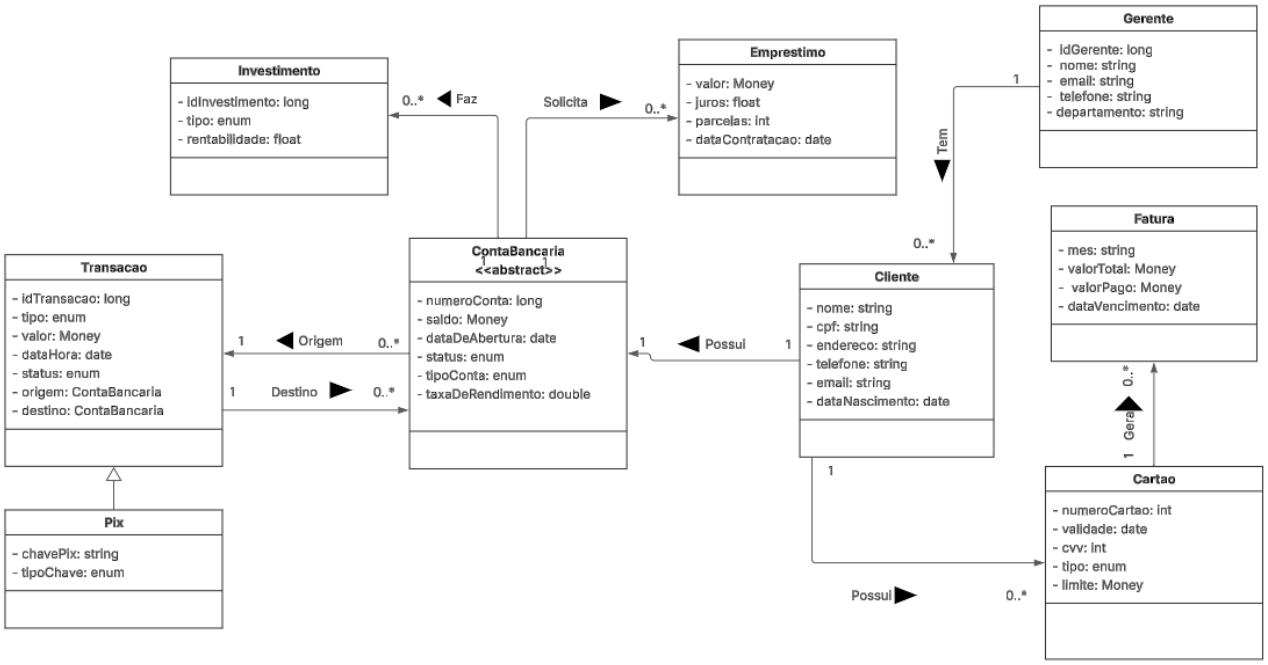
UC05 - Solicitar cartão	
Descrição/Objetivo	O cliente solicita um cartão de crédito ou débito.
Tipo de UC	Primário
Atores	Cliente (Primário) e gerente (Secundário). .
Pré-Condições	● O cliente deve ter uma conta ativa e estar logado no sistema.

Pós-Condições:	● A solicitação de cartão é registrada no sistema. ● Se aprovado, o cartão é emitido e enviado ao endereço do cliente.
Trigger:	
Fluxo Básico:	1. O cliente acessa a área de solicitação de cartões. 2. O cliente escolhe o tipo de cartão desejado (débito, crédito ou múltiplo). 3. O sistema exibe os termos e condições de uso do cartão. 4. O cliente lê e aceita os termos. 5. O sistema confirma o endereço de entrega (baseado no cadastro). 6. O cliente finaliza a solicitação. 7. O sistema registra o pedido e o envia para análise do gerente, informando que a solicitação será processada em horário comercial (RF21). 8. O sistema exibe uma mensagem de confirmação com o prazo estimado de entrega (RN13).
Fluxos Alternativos:	● **FA01 - Alterar endereço de entrega (inicia no Passo 5)**: O cliente pode optar por alterar o endereço de entrega antes de finalizar a solicitação.
Fluxos de Exceções:	● **FE01 - Solicitação negada (inicia no Passo 7)** : Se a análise de crédito (para cartão de crédito) for negativa, o sistema informa a recusa ao cliente.
Referências Cruzadas (Requisitos Associados):	RN13, RF21.

UC06 - Excluir conta	
Descrição/Objetivo	Permitir que um cliente encerre sua conta bancária e o vínculo com o banco Paper.
Tipo de UC	Secundário
Atores	Cliente (Primário) e Gerente (Secundário).
Pré-Condições	● O usuário não pode possuir uma conta existente vinculada ao seu CPF. ● O usuário deve ter acesso à internet e possuir um e-mail e telefone válidos.
Pós-Condições:	● A conta do cliente é criada e ativada no sistema. ● Um número de agência e conta é gerado e vinculado ao cadastro.

	O cliente é associado a um gerente exclusivo (RN20). O histórico do cliente começa a ser registrado.
Trigger:	
Fluxo Básico:	1. O cliente inicia o processo de encerramento de conta. 2. O sistema verifica a existência de pendências financeiras (RF27). 3. Não havendo pendências, o sistema exibe um resumo das implicações do encerramento e solicita uma confirmação final. 4. Para confirmar, o sistema solicita a senha de acesso do cliente. 5. O cliente insere a senha e confirma a exclusão. 6. O sistema desativa a conta, revoga o acesso e notifica o gerente responsável. 7. O sistema exibe uma mensagem final confirmando que a conta foi encerrada.
Fluxos Alternativos:	N/A
Fluxos de Exceções:	● **FE01 - Pendências financeiras encontradas (inicia no Passo 2)** : Se o sistema detectar pendências (saldo negativo, dívidas), o processo é bloqueado, e uma mensagem é exibida orientando o cliente a regularizar sua situação antes de prosseguir.
Referências Cruzadas (Requisitos Associados):	RN24, RF27.

6. Diagrama de Classes



6.1. Descrição das Entidades

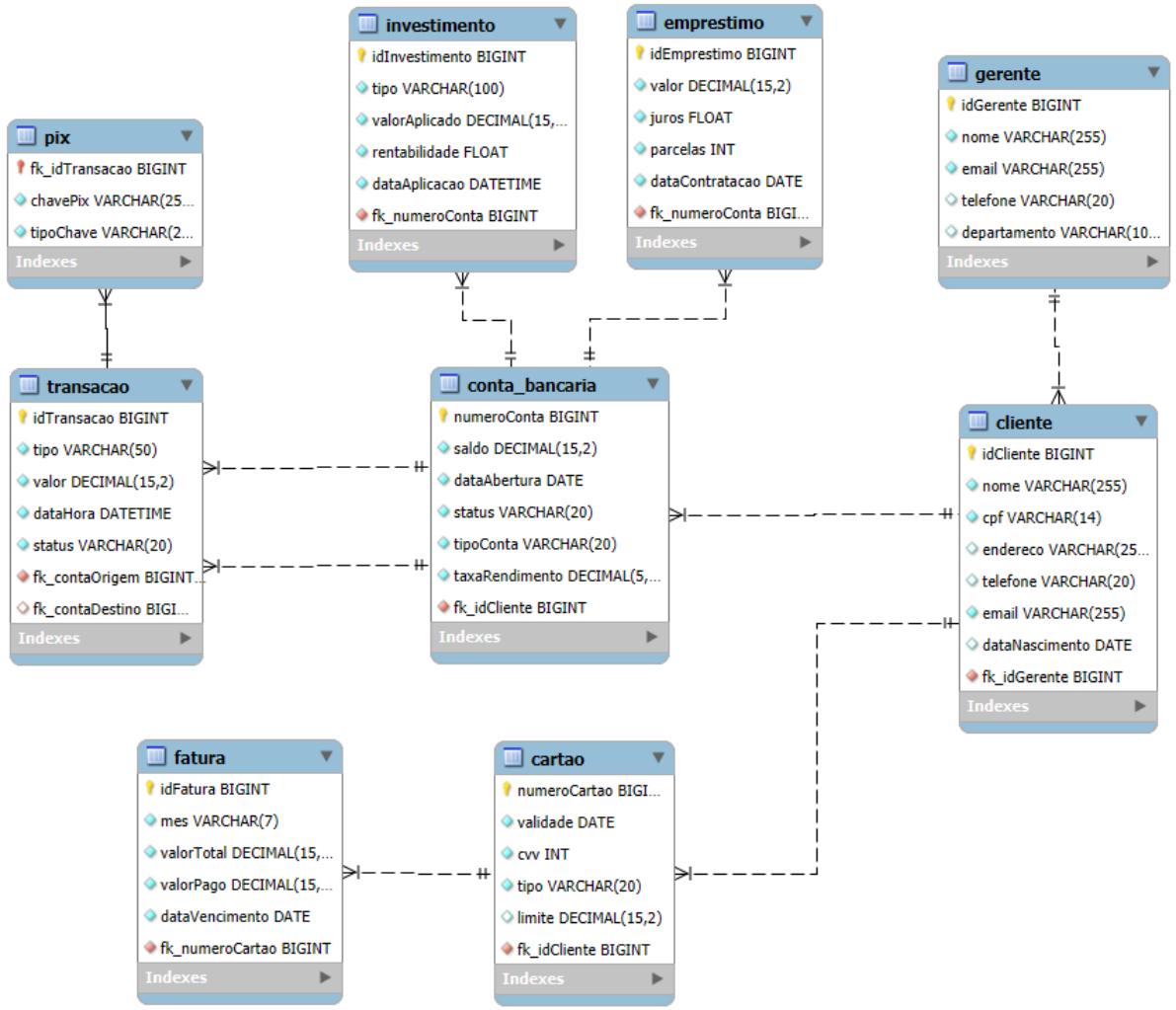
As classes serão explicadas na tabela a seguir:

Classe	Atributos	Descrição
Cliente	- nome: string - cpf: string - endereco: string - telefone: string - email: string - dataNascimento: date	Esta classe serve para cada cliente do banco. Cada cliente que criar uma conta vai ter uma instância da classe na qual os dados serão guardados no banco de dados.
ContaBancaria	- numeroConta: long - saldo: Money - dataDeAbertura: date - status: enum	Esta classe é abstrata e é responsável por garantir que as classes ContaCorrente e ContaPoupanca armazenem os dados e realizem as ações referente ao banco, como ver o saldo e realizar transações bancárias.
Transacao	- idTransacao: long - tipo: enum - valor: Money - dataHora: date - status: enum	Responsável por criar, monitorar e realizar ações determinadas as transações bancárias
Pix	(Herda da classe Transacao) - chavePix: string - tipoChave: enum	É uma especificação da classe Transacao e é responsável por transferências do tipo pix.
ContaCorrente	(Herda da classe ContaBancaria)	É uma especificação da classe ContaBancaria e define a conta do cliente como conta corrente.
ContaPoupanca	(Herda da classe ContaBancaria) - taxaRendimento: float	É uma especificação da classe ContaBancaria e define a conta do cliente como conta poupança.
Cartao	- numeroCartao: int - validade: date - cvv: int - tipo: enum - limite: Money	Responsável por guardar, criar e monitorar os dados do cartão do cliente.
Fatura	- mes: string - valorTotal: Money - valorPago: Money - dataVencimento: date	Esta classe é responsável pela fatura gerada pelo cartão de crédito.

Investimento	- idInvestimento: long - tipo: enum - rentabilidade: float	É responsável por monitorar os investimentos do cliente.
Emprestimo	- valor: Money - juros: float - parcelas: int - dataContratacao: date	É responsável por monitorar os empréstimos do cliente.
Gerente	- idGerente: long - nome: string - email: string - telefone: string - departamento: string	

7. Banco de Dados

7.1. Esquema Lógico de Banco de Dados Relacional



7.2. Descrições

Entidade	Descrição
ContaBancaria	Representa a conta bancária do cliente. Pode ser do tipo corrente ou poupança. Está associada diretamente ao cliente.
Cartão	Armazena os dados dos cartões vinculados ao cliente, podendo ser de débito, crédito ou múltiplo.
Fatura	Representa as faturas mensais geradas pelos cartões de crédito, com valores totais, pagos e data de vencimento.
Empréstimo	Armazena os empréstimos contratados pelo cliente, com valor, juros, número de parcelas e data de contratação.

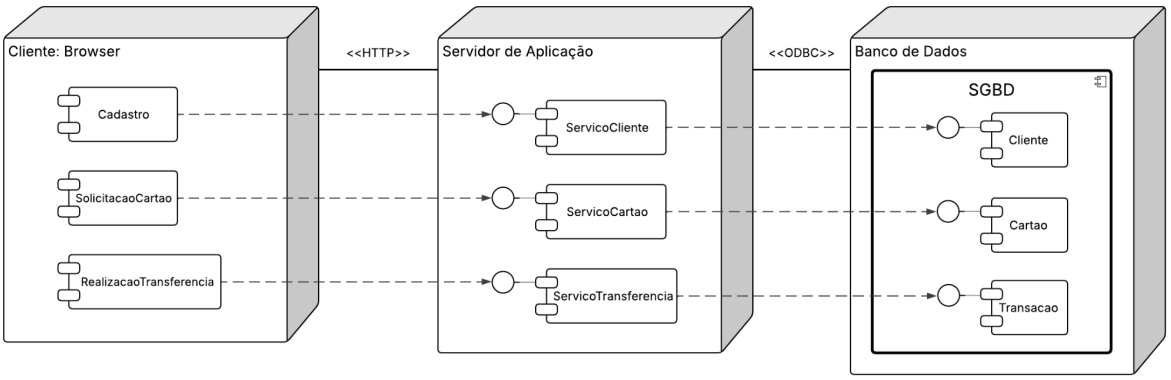
7.3. Observações

Referência Cruzada com Requisitos:
O modelo foi construído com base nos requisitos funcionais e não funcionais descritos no documento, como:

- RN01 (CPF único por cliente)
- RN20 (cliente vinculado a gerente)
- RF04 (conta vinculada ao cliente)
- RF07 (autenticação em transações)
- RNF04 (auditoria de transações)

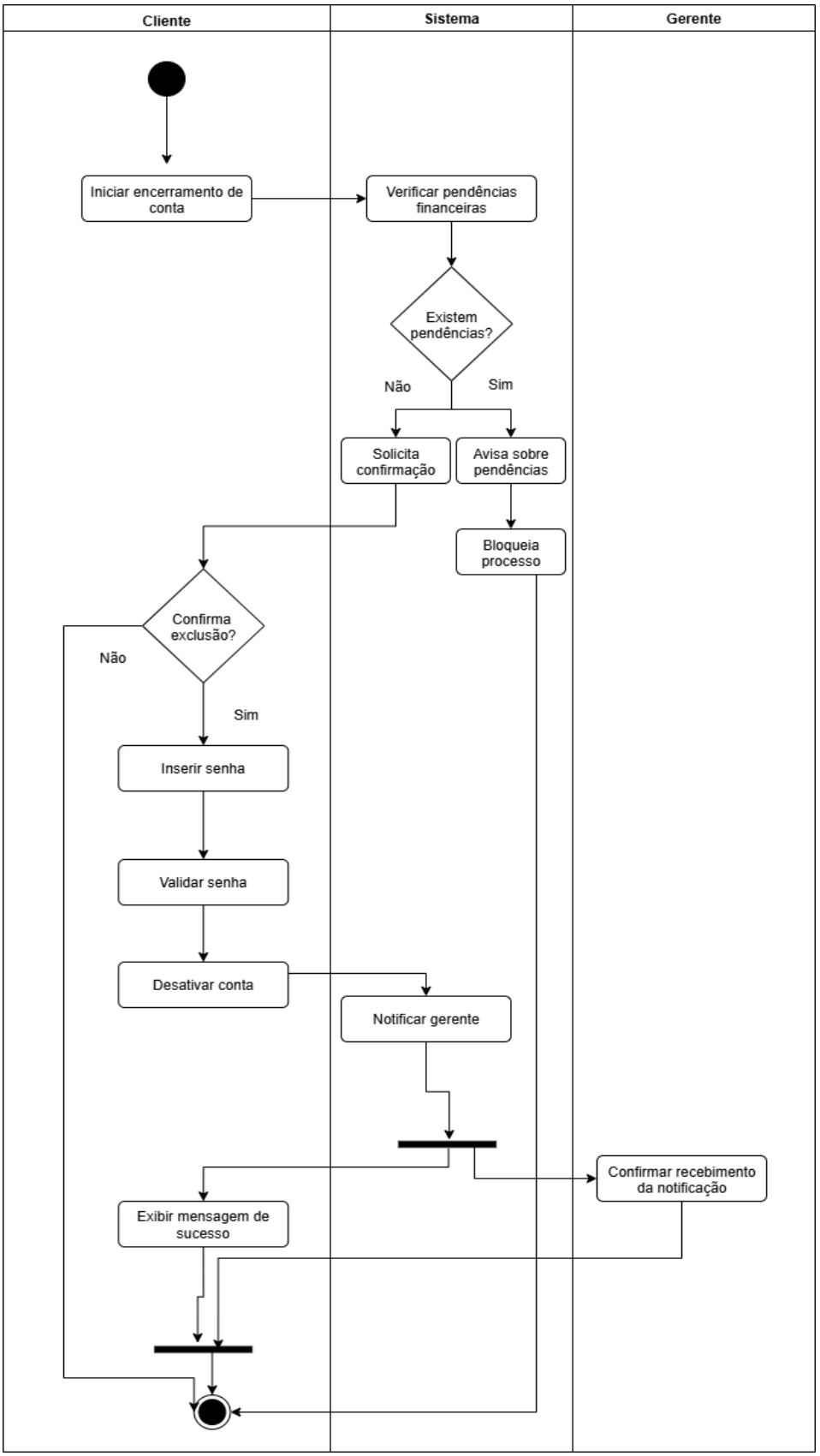
8. Projeto Arquitetural do Sistema

8.1. Diagrama de Alocação de Recursos

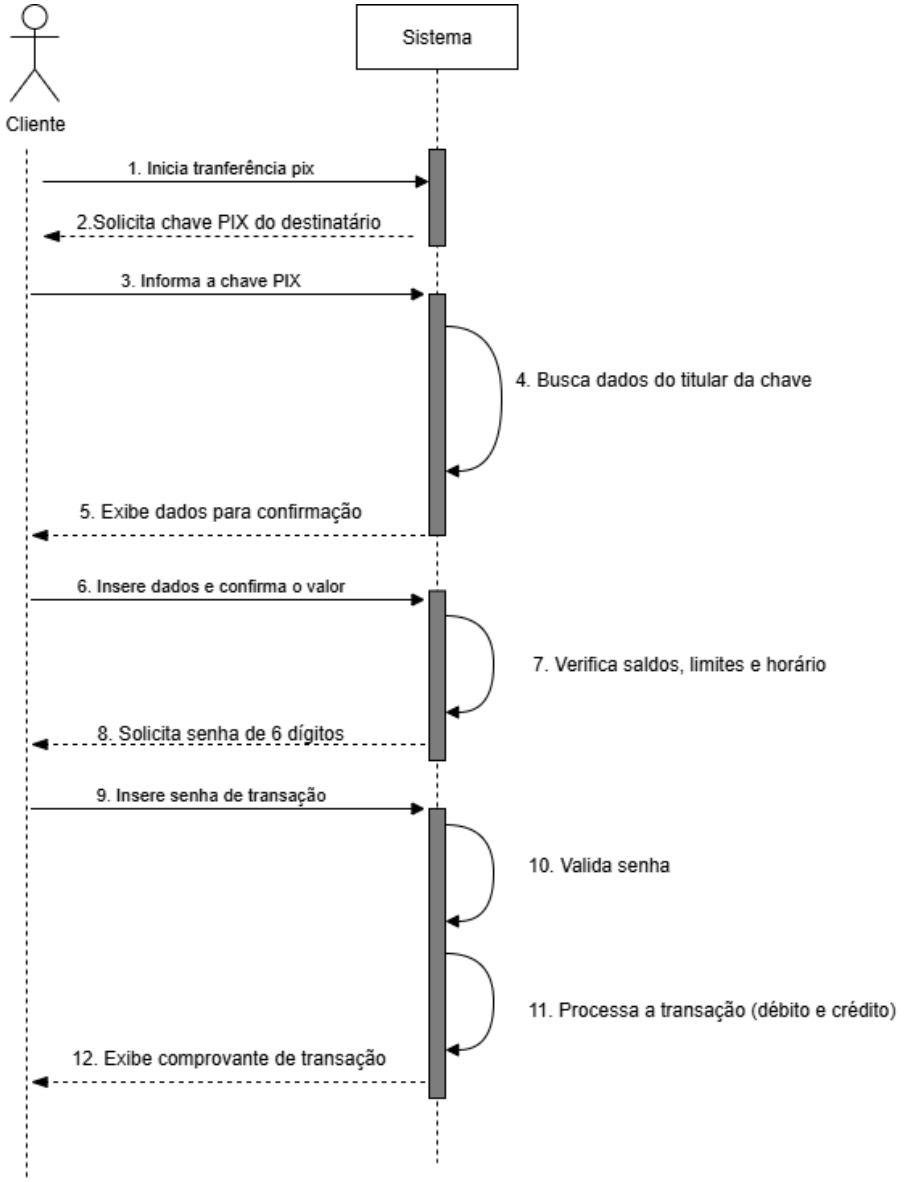


9. Projeto Funcional do Sistema

9.1. Diagrama de Atividades

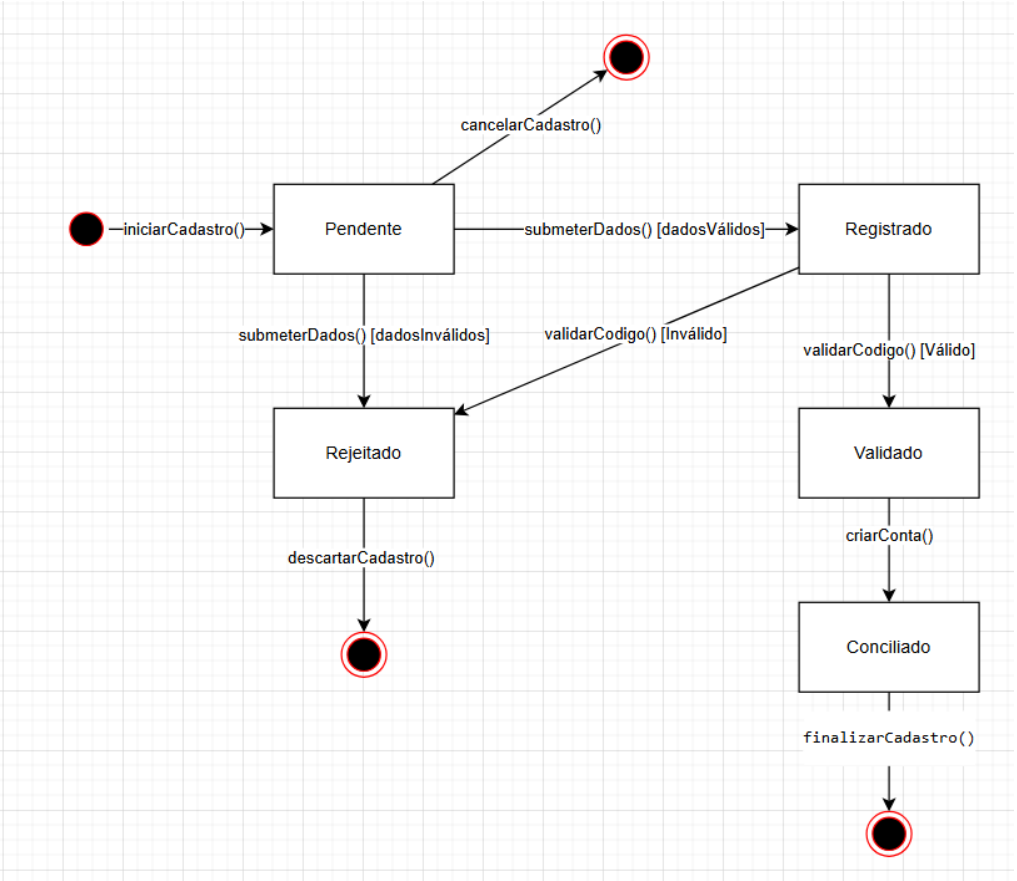


9.2. Diagrama de Sequência



9.3. Diagrama de Estados

UC01 - Realizar Cadastro - Objeto: Cadastro



10. Banco de Dados Relacional

10.1. Criação do Banco de Dados (DDL)

SGBD: MariaDB **Versão:** 10.4.32

```
DROP DATABASE IF EXISTS paperbank;

CREATE DATABASE paperbank CHARACTER SET utf8mb4 COLLATE
utf8mb4_unicode_ci;

USE paperbank;

CREATE TABLE gerente (
  idGerente BIGINT NOT NULL AUTO_INCREMENT,
  nome VARCHAR(255) NOT NULL,
  email VARCHAR(255) NOT NULL UNIQUE,
  telefone VARCHAR(20),
```

```
  departamento VARCHAR(100) NOT NULL,
  PRIMARY KEY (idGerente)
);

CREATE TABLE cliente (
  idCliente BIGINT NOT NULL AUTO_INCREMENT,
  nome VARCHAR(255) NOT NULL,
  cpf VARCHAR(14) NOT NULL UNIQUE,
      senha VARCHAR(255) NOT NULL,
  endereco VARCHAR(255),
  telefone VARCHAR(20),
  email VARCHAR(255) NOT NULL UNIQUE,
  dataNascimento DATE,
  fk_idGerente BIGINT NOT NULL,
  PRIMARY KEY (idCliente),
  FOREIGN KEY (fk_idGerente) REFERENCES gerente (idGerente)
    ON DELETE RESTRICT
    ON UPDATE CASCADE
);

CREATE TABLE conta_bancaria (
  numeroConta BIGINT NOT NULL,
  saldo DECIMAL(15,2) NOT NULL DEFAULT 0.00,
  dataAbertura DATE NOT NULL,
  status VARCHAR(20) NOT NULL,
  tipoConta VARCHAR(20) NOT NULL,
  taxaRendimento DECIMAL(5,2),
  fk_idCliente BIGINT NOT NULL,
  PRIMARY KEY (numeroConta),
  FOREIGN KEY (fk_idCliente) REFERENCES cliente (idCliente)
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

CREATE TABLE cartao (
  numeroCartao BIGINT NOT NULL,
  validade DATE NOT NULL,
  cvv INT NOT NULL,
  tipo VARCHAR(20) NOT NULL,
  limite DECIMAL(15,2) NOT NULL,
  fk_idCliente BIGINT NOT NULL,
  PRIMARY KEY (numeroCartao),
  FOREIGN KEY (fk_idCliente) REFERENCES cliente (idCliente)
```

```
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

CREATE TABLE fatura (
  idFatura BIGINT NOT NULL AUTO_INCREMENT,
  mes VARCHAR(7) NOT NULL, -- Formato YYYY-MM
  valorTotal DECIMAL(15,2) NOT NULL,
  valorPago DECIMAL(15,2) DEFAULT 0.00,
  dataVencimento DATE NOT NULL,
  fk_numeroCartao BIGINT NOT NULL,
  PRIMARY KEY (idFatura),
  FOREIGN KEY (fk_numeroCartao) REFERENCES cartao (numeroCartao)
    ON DELETE RESTRICT
    ON UPDATE CASCADE
);

CREATE TABLE investimento (
  idInvestimento BIGINT NOT NULL AUTO_INCREMENT,
  tipo VARCHAR(100) NOT NULL,
  valorAplicado DECIMAL(15,2) NOT NULL,
  rentabilidade FLOAT,
  dataAplicacao DATETIME NOT NULL,
  fk_numeroConta BIGINT NOT NULL,
  PRIMARY KEY (idInvestimento),
  FOREIGN KEY (fk_numeroConta) REFERENCES conta_bancaria (numeroConta)
    ON DELETE RESTRICT
    ON UPDATE CASCADE
);

CREATE TABLE emprestimo (
  idEmprestimo BIGINT NOT NULL AUTO_INCREMENT,
  valor DECIMAL(15,2) NOT NULL,
  juros FLOAT NOT NULL,
  parcelas INT NOT NULL,
  dataContratacao DATE NOT NULL,
  fk_numeroConta BIGINT NOT NULL,
  PRIMARY KEY (idEmprestimo),
  FOREIGN KEY (fk_numeroConta) REFERENCES conta_bancaria (numeroConta)
    ON DELETE RESTRICT
    ON UPDATE CASCADE
);
```

```sql
CREATE TABLE transacao (
  idTransacao BIGINT NOT NULL AUTO_INCREMENT,
  tipo VARCHAR(50) NOT NULL,
  valor DECIMAL(15,2) NOT NULL,
  dataHora DATETIME NOT NULL,
  status VARCHAR(20) NOT NULL,
  fk_contaOrigem BIGINT,
  fk_contaDestino BIGINT,
  PRIMARY KEY (idTransacao),
  FOREIGN KEY (fk_contaOrigem) REFERENCES conta_bancaria (numeroConta),
  FOREIGN KEY (fk_contaDestino) REFERENCES conta_bancaria (numeroConta)
);

CREATE TABLE pix (
  idPixTransacao BIGINT NOT NULL,
  chavePix VARCHAR(255) NOT NULL,
  tipoChave VARCHAR(20) NOT NULL,
  PRIMARY KEY (idPixTransacao),
  FOREIGN KEY (idPixTransacao) REFERENCES transacao (idTransacao)
    ON DELETE CASCADE
    ON UPDATE CASCADE
);
```

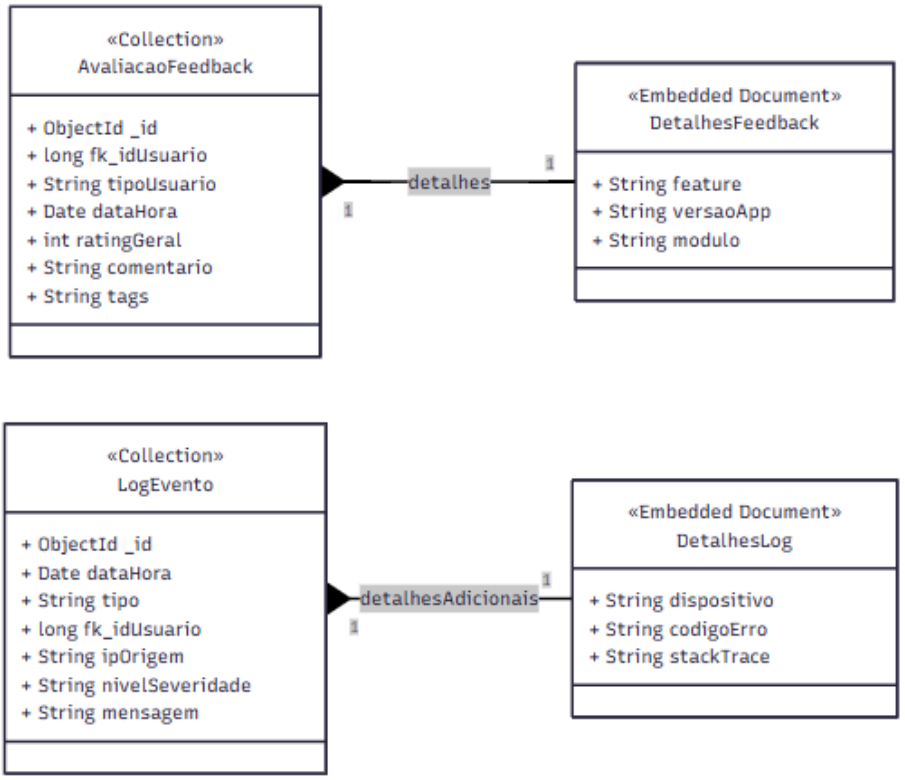
10.2. População Inicial do Banco (DML)

```sql
USE paperbank;

INSERT INTO gerente (nome, email, telefone, departamento) VALUES
('Carlos Silva', 'carlos.silva@paperbank.com', '21987654321', 'Contas
Corporativas'),
('Ana Pereira', 'ana.pereira@paperbank.com', '21988776655',
'Investimentos'),
('Roberto Souza', 'roberto.souza@paperbank.com', '21999887766',
'Crédito Pessoal');

INSERT INTO cliente (nome, cpf, endereco, telefone, email,
dataNascimento, fk_idGerente) VALUES
('João da Silva', '111.222.333-44', 'Rua das Flores, 123',
'21911112222', 'joao.silva@email.com', '1985-05-10', 1),
('Maria Oliveira', '555.666.777-88', 'Avenida Principal, 456',
'21933334444', 'maria.oliveira@email.com', '1990-11-20', 2),
```

```
('Pedro Santos', '999.888.777-66', 'Travessa dos Coqueiros, 789',
'21955556666', 'pedro.santos@email.com', '1988-01-30', 1);

INSERT INTO conta_bancaria (numeroConta, saldo, dataAbertura, status,
tipoConta, taxaRendimento, fk_idCliente) VALUES
(1001, 5000.75, '2020-01-15', 'ATIVA', 'Corrente', NULL, 1),
(2002, 15250.00, '2019-03-22', 'ATIVA', 'Poupança', 0.05, 2),
(1003, 850.50, '2021-07-30', 'ATIVA', 'Corrente', NULL, 3);

INSERT INTO cartao (numeroCartao, validade, cvv, tipo, limite,
fk_idCliente) VALUES
(1234567890123456, '2028-12-31', 123, 'Crédito', 4000.00, 1),
(9876543210987654, '2027-10-31', 456, 'Crédito', 10000.00, 2),
(1122334455667788, '2029-05-31', 789, 'Débito', 0.00, 3);

INSERT INTO fatura (mes, valorTotal, valorPago, dataVencimento,
fk_numeroCartao) VALUES
('2025-10', 1250.50, 1250.50, '2025-10-10', 1234567890123456),
('2025-09', 850.00, 850.00, '2025-09-10', 1234567890123456),
('2025-10', 3500.00, 0.00, '2025-10-12', 9876543210987654);

INSERT INTO investimento (tipo, valorAplicado, rentabilidade,
dataAplicacao, fk_numeroConta) VALUES
('CDB Pós-Fixado', 10000.00, 0.11, '2023-01-20 10:00:00', 2002),
('Ações PETR4', 5000.00, 0.15, '2023-05-15 14:30:00', 2002);

INSERT INTO emprestimo (valor, juros, parcelas, dataContratacao,
fk_numeroConta) VALUES
(15000.00, 0.08, 24, '2022-08-01', 1001);

INSERT INTO transacao (tipo, valor, dataHora, status, fk_contaOrigem,
fk_contaDestino) VALUES
('Transferência', 500.00, '2025-10-17 10:30:00', 'Concluída', 1001,
2002),
('Depósito', 1000.00, '2025-10-16 15:00:00', 'Concluída', NULL, 1003),
('PIX Enviado', 150.00, '2025-10-18 09:00:00', 'Concluída', 2002,
NULL),
('Saque', 200.00, '2025-10-15 12:45:00', 'Concluída', 1003, NULL);

INSERT INTO pix (idPixTransacao, chavePix, tipoChave) VALUES
(3, 'pedro.santos@email.com', 'Email');
```

11. Banco de Dados Não-Convencional

11.1. Diagrama do Banco NOSQL



11.2. Criação e População do Banco de Dados

SGBD: MongoDB **Versão:** 8.2

```
db.avaliacoes_feedback.insertMany([
    {
        idFeedback: 1,
        dataHora: new Date("2025-10-15T14:45:00Z"),
        tipoUsuario: "Cliente",
        fk_idUsuario: 1,
        ratingGeral: 4,
        comentario: "O novo layout para transferências PIX ficou ótimo,
mas o app está demorando para carregar o extrato da poupança.",
        tags: ["Sugestão", "Lentidão", "UI/UX"],
        detalhes: {
            feature: "Transferência PIX",
            versaoApp: "2.1.0"
```

```
        }
    },
    {
        idFeedback: 2,
        dataHora: new Date("2025-10-18T09:10:00Z"),
        tipoUsuario: "Gerente",
        fk_idUsuario: 2,
        ratingGeral: 5,
        comentario: "A ferramenta de visualização de clientes
inadimplentes está muito ágil!",
        tags: ["Melhoria", "Produtividade"],
        detalhes: {
            modulo: "Backoffice - Crédito",
            sistema: "Web Gerencial"
        }
    }
]);

db.logs_eventos.insertMany([
    {
        dataHora: new Date("2025-10-23T16:00:00Z"),
        tipo: "Login",
        fk_idUsuario: 1,
        ipOrigem: "200.145.10.1",
        nivelSeveridade: "INFO",
        mensagem: "Login de cliente bem-sucedido.",
        detalhesAdicionais: {
            dispositivo: "Android App",
            sessaoId: "XYZ123ABC"
        }
    },
    {
        dataHora: new Date("2025-10-23T16:05:30Z"),
        tipo: "Erro",
        fk_idUsuario: 2,
        ipOrigem: "192.168.1.5",
        nivelSeveridade: "ERROR",
        mensagem: "Falha ao carregar lista de clientes para análise de
crédito.",
        detalhesAdicionais: {
            codigoErro: "DB-005",
            stackTrace: "Erro de timeout na API de Clientes..."
        }
    }
]);
```

```
db.logs_eventos.createIndex(
    { "dataHora": 1 },
    { expireAfterSeconds: 7776000 }
);
```

12. Implementação

https://github.com/thauanhub/paper-bank

13. Testes de Carga

13.1. 1ª Fase de Testes

13.1.1. OBJETIVOS DO PROJETO

13.1.1.1. OBJETIVO GERAL

- Realizar a análise de desempenho de serviços internos do sistema através de testes de carga, identificando métricas críticas de performance e propondo hipóteses para possíveis gargalos.

13.1.1.2. OBJETIVOS ESPECÍFICOS

- Selecionar um conjunto mínimo de dois serviços internos para análise, garantindo que pelo menos um deles execute operações de escrita no banco de dados.
- Projetar e executar testes de carga nos serviços selecionados, submetendo-os a diferentes cenários de demanda.
- Coletar e analisar métricas de desempenho durante os testes, incluindo:
    - Latência (tempo médio de resposta)
    - Vazão (requisições processadas por segundo em intervalos específicos: 1s, 5s, 10s, 30s e 1min)
    - Capacidade de concorrência (número máximo de requisições simultâneas suportadas)

13.1.2. RESULTADO DO TERMINAL

```
  execution: local
     script: index.js
     output: json (resultado.json)

  scenarios: (100.00%) 1 scenario, 10 max VUs, 1m0s max duration (incl. graceful stop):
           * default: 10 looping VUs for 30s (gracefulStop: 30s)


TOTAL RESULTS

  checks_total.......: 392     12.325396/s
  checks_succeeded...: 100.00% 392 out of 392
  checks_failed......: 0.00%   0 out of 392

  ✓ GET /saldo - 200
  ✓ POST /registrar - 201

  CUSTOM
  concorrencia_obter_saldo.........: 1      min=1        max=1
  concorrencia_registrar_cliente...: 1      min=1        max=1
  latencia_obter_saldo.............: avg=30.375829  min=5.8346   med=10.55275 max=330.6775 p(90)=60.82755 p(95)=129.712775
  latencia_registrar_cliente.......: avg=543.154458 min=381.3508 med=536.9724 max=814.1836 p(90)=631.2235 p(95)=713.834
  vazao_obter_saldo................: 196    6.162698/s
  vazao_registrar_cliente..........: 196    6.162698/s

  HTTP
  http_req_duration.................: avg=287.11ms   min=5.83ms   med=381.35ms max=814.18ms p(90)=578.55ms p(95)=630.05ms
    { expected_response:true }.....: avg=287.11ms   min=5.83ms   med=381.35ms max=814.18ms p(90)=578.55ms p(95)=630.05ms
  http_req_failed..................: 0.00%  0 out of 393
  http_reqs........................: 393    12.356838/s

  EXECUTION
  iteration_duration...............: avg=1.57s      min=1.39s    med=1.54s    max=2.09s    p(90)=1.67s    p(95)=1.81s
  iterations.......................: 196    6.162698/s
  vus..............................: 8      min=8        max=10
  vus_max..........................: 10     min=10       max=10

  NETWORK
  data_received....................: 83 kB  2.6 kB/s
```

13.1.3. MEDIÇÕES DO SLA

a) Serviço: Obter Saldo (ObterSaldo.js)

- Tipo de operação: Leitura
- Arquivos envolvidos: https://github.com/thauanhub/paper-bank/blob/main/backend/auth.py
- Arquivos de medição SLA: https://github.com/GbosDev/TestesDeCargaK6-PaperBank.git
- Data da medição: 23/11/2025
- Configurações: 12th Gen Intel(R) Core(TM) i5-1235U (1.30 GHz), 16,0 GB, 64bit, Node.js v25.2.1, Banco MySQL + MongoDB

- Testes de carga:

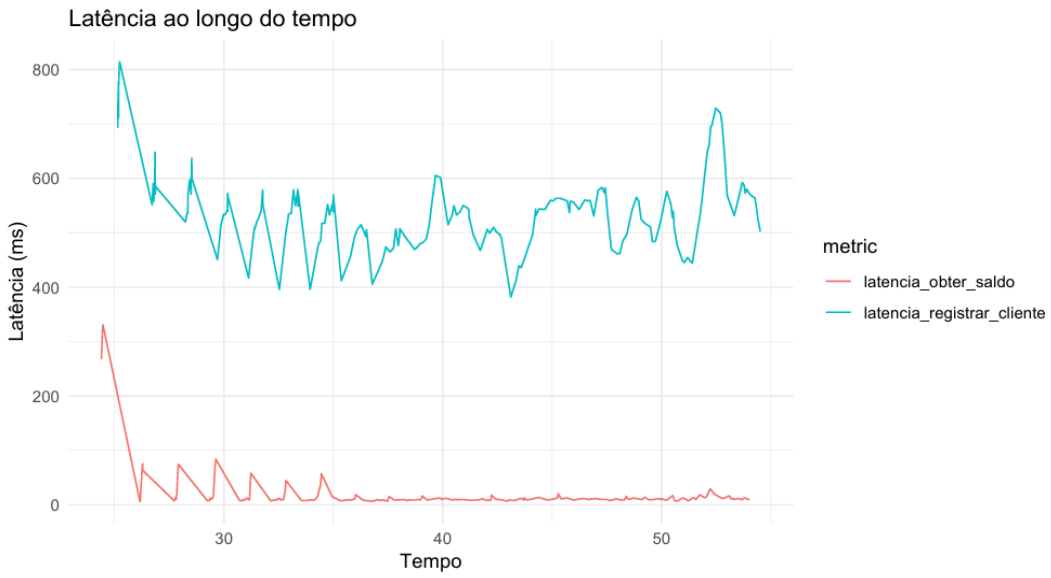
- Latência: 30.37 ms (média), 129.71 ms (p95)
- Vazão: 6.16 requisições/segundo
- Concorrência: 1 requisição simultânea por usuário virtual

b) Serviço: Registrar Cliente (RegistrarCliente.js)

- Tipo de operação: Inserção
- Arquivos envolvidos: https://github.com/thauanhub/paper-bank/blob/main/backend/auth.py
- Arquivos de medição SLA: https://github.com/GbosDev/TestesDeCargaK6-PaperBank.git
- Data da medição: 23/11/2025
- Configurações: 12th Gen Intel(R) Core(TM) i5-1235U (1.30 GHz), 16,0 GB, 64bit, Node.js v25.2.1, Banco MySQL + MongoDB
- Testes de carga:
  - Latência: 543.15 ms (média), 713.83 ms (p95)
  - Vazão: 6.16 requisições/segundo
  - Concorrência: 1 requisição simultânea por usuário virtual

13.1.4. ANÁLISE DOS RESULTADOS DO TESTE DE CARGA

13.1.4.1. Gráfico de Latência × Tempo

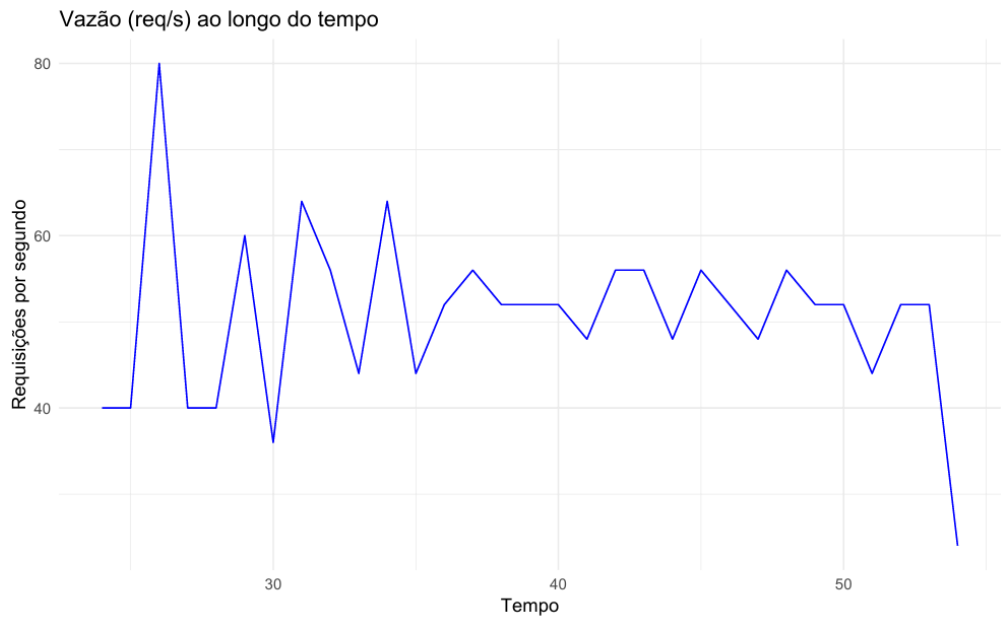


- Para este cenário foram feitas as seguintes análises:

O gráfico demonstra uma diferença significativa no desempenho entre as duas operações analisadas. A consulta de saldo (GET /saldo) manteve-se consistentemente rápida e estável, com tempo médio de resposta de 30,38 ms. Em contraste, a operação de registro (POST /registrar) apresentou tempo médio consideravelmente superior (543,15 ms), com variações mais acentuadas durante o teste.

Este comportamento era esperado, uma vez que operações de escrita tendem a ser naturalmente mais lentas que operações de leitura.

13.1.4.2. Gráfico de Vazão (Requisição por segundo) X Tempo

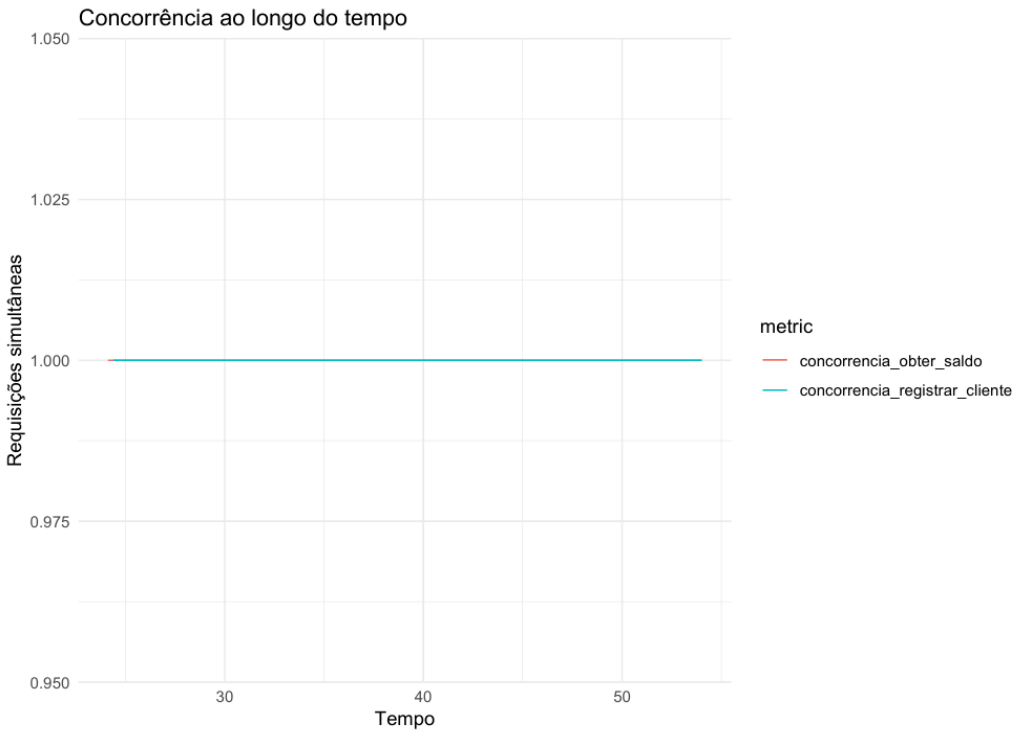


- Para este cenário foram feitas as seguintes análises:

Embora a média de requisições por segundo tenha sido de 12,36, o gráfico revela uma instabilidade considerável neste indicador, com flutuações frequentes ao longo do tempo. Essas variações sugerem que o sistema enfrenta dificuldades para manter um throughput constante, possivelmente devido a limitações na conexão com o banco de dados ou na alocação de recursos do servidor.

Tal instabilidade merece atenção, pois pode impactar diretamente a experiência do usuário final.

13.1.4.3. Gráfico de Concorrência X Tempo



- Para este cenário foram feitas as seguintes análises:

O gráfico de concorrência apresentou um comportamento estável para ambas as operações, mantendo-se em aproximadamente 1 requisição simultânea processada por vez. Este padrão indica que o sistema gerencia adequadamente a distribuição do processamento entre operações rápidas e lentas, sem acumular requisições em excesso.

13.1.5. CONCLUSÃO E IDENTIFICAÇÃO DE PONTOS CRÍTICOS

Os testes confirmaram a confiabilidade do sistema, com 100% das requisições processadas com sucesso. Entretanto, a análise identificou aspectos que requerem melhorias:

- Operações de registro com desempenho limitado: O tempo de resposta elevado no endpoint de registro indica possíveis gargalos, provavelmente relacionados a transações de banco de dados ou processos de validação.

- Instabilidade na vazão: A flutuação no número de requisições por segundo sugere dificuldades do sistema em manter performance consistente sob carga contínua.

- Capacidade limitada de processamento simultâneo: A baixa concorrência mantida durante os testes pode indicar restrições na arquitetura para aproveitar todo o potencial de processamento paralelo.

Próximas prioridades levantadas:

1. Otimizar as operações de registro, com foco nas transações de banco de dados
2. Revisar a configuração de pools de conexão e recursos compartilhados
3. Avaliar a implementação de processamento assíncrono para operações mais demoradas
4. Realizar testes de estresse para determinar os limites de escalabilidade do sistema

Em resumo, o sistema apresenta base sólida de funcionamento, mas requer ajustes específicos para melhorar o desempenho das operações de escrita e garantir uma vazão mais estável sob carga.

13.2. 2ª Fase de Testes

13.2.1. OBJETIVO GERAL

Esta demonstra o segundo teste de carga a fim de comparação entre os resultados de dois testes realizados sequencialmente no sistema. O 1° teste representa a configuração inicial, enquanto o 2° teste foi executado após a implementação de otimizações de performance e a adição de um novo endpoint (DELETE /conta/excluir).

A comparação direta dos dados permite avaliar o impacto das melhorias técnicas na capacidade de resposta e estabilidade do sistema sob carga. Ambos os testes mantiveram uma taxa de sucesso de 100% em todas as requisições, demonstrando a robustez da aplicação mesmo com a expansão de funcionalidades.

Os resultados a seguir detalham a análise dos ajustes, focando em métricas críticas como latência, vazão e concorrência entre os dois cenários.

13.2.2. DESCRIÇÃO DAS OTIMIZAÇÕES

1. Otimização da Consulta de Saldo:

- Implementação de query seletiva utilizando load_only para retornar exclusivamente as colunas necessárias, reduzindo a complexidade computacional
- Remoção da conversão redundante do saldo para float(), preservando a integridade dos dados no formato original

2. Otimização do Processo de Registro:

- Implementação de processamento assíncrono de logs mediante a integração da biblioteca BackgroundTasks, permitindo que a geração de logs ocorra através de função auxiliar após a confirmação de sucesso, sem bloquear requisições subsequentes
- Unificação das operações de banco de dados relacionadas às entidades Cliente e Conta, substituindo múltiplas operações de commit por db.flush() para garantir a geração de IDs, com um único db.commit() final para minimizar operações de escrita em disco

13.2.2. RESULTADO DAS MEDIÇÕES COMPARATIVAMENTE

13.2.2.1. RESULTADO NO TERMINAL

```
INFO[0000] Criando 200 clientes                          source=console
INFO[0148] Total de usuários criados: 200                source=console
INFO[0179] Iniciando exclusão única de todas as contas   source=console
INFO[0255] Todas as contas foram excluídas.              source=console


  TOTAL RESULTS

    checks_total.......: 596     2.336253/s
    checks_succeeded...: 100.00% 596 out of 596
    checks_failed......: 0.00%   0 out of 596

    ✓ GET /saldo - 200
    ✓ POST /registrar - 201
    ✓ DELETE /conta/excluir - sucesso no teardown

    CUSTOM
    concorrencia_excluir_conta.......: 1      min=1        max=1
    concorrencia_obter_saldo.........: 1      min=1        max=1
    concorrencia_registrar_cliente...: 1      min=1        max=1
    latencia_excluir_conta...........: avg=378.779039 min=301.9985 med=365.21635 max=618.198   p(90)=462.13718 p(95)=492.52839
    latencia_obter_saldo.............: avg=30.11569   min=5.2097   med=9.9647    max=293.9618  p(90)=94.59156  p(95)=120.100195
    latencia_registrar_cliente.......: avg=520.851563 min=377.8662 med=459.57005 max=1023.3192 p(90)=684.34061 p(95)=799.906325
    vazao_excluir_conta..............: 200    0.783978/s
    vazao_obter_saldo................: 198    0.776138/s
    vazao_registrar_cliente..........: 198    0.776138/s

    HTTP
    http_req_duration.................: avg=334.18ms   min=5.2ms    med=359.21ms  max=1.02s     p(90)=507.3ms   p(95)=573.46ms
      { expected_response:true }.....: avg=334.18ms   min=5.2ms    med=359.21ms  max=1.02s     p(90)=507.3ms   p(95)=573.46ms
    http_req_failed...................: 0.00%  0 out of 996
    http_reqs.........................: 996    3.904209/s

    EXECUTION
    iteration_duration................: avg=1.55s      min=1.38s    med=1.46s     max=2.13s     p(90)=1.84s     p(95)=1.93s
    iterations........................: 198    0.776138/s
    vus...............................: 0      min=0        max=10
    vus_max...........................: 10     min=10       max=10

    NETWORK
    data_received.....................: 234 kB 916 B/s
    data_sent.........................: 280 kB 1.1 kB/s
```

13.2.2.2. MEDIÇÕES DO SLA

a) Serviço: Obter Saldo (ObterSaldo.js)

- **Tipo de operações: leitura**
- **Arquivos envolvidos**
  https://github.com/thauanhub/paper-bank/blob/main/backend/auth.py
- **Arquivos com o código fonte de medição do SLA:**
  https://github.com/GbosDev/TestesDeCargaK6-PaperBank.git
- Configurações: 12th Gen Intel(R) Core(TM) i5-1235U (1.30 GHz), 16,0 GB, 64bit, Node.js v25.2.1, Banco MySQL + MongoDB

- **MEDIÇÃO 1**
- **Data da medição: 23/11/2025**
- **Testes de carga (SLA):**
  - Latência: 30.37 ms (média), 129.71 ms (p95)
  - Vazão: 6.16 requisições/segundo
  - Concorrência: 1 requisição simultânea por usuário virtual

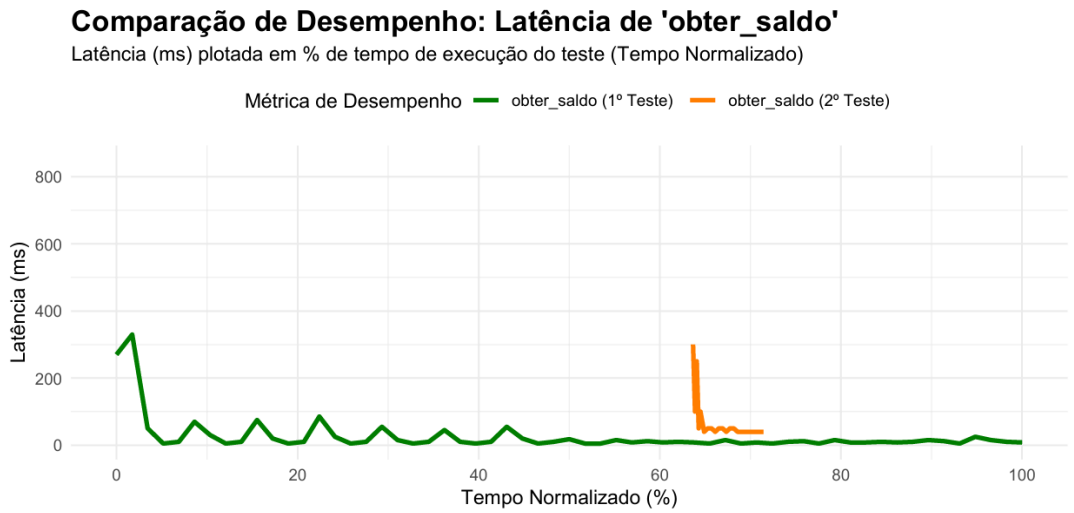
- **Potenciais gargalos do sistema**

  Apesar da operação ser leve e baseada em leitura, a MEDIÇÃO 1 apresenta um p95 elevado (129 ms) em relação à média (~30 ms), indicando que parte das requisições sofre atrasos ocasionais. Esses picos sugerem possíveis gargalos relacionados a:

  1) Consultas não otimizadas, retornando mais dados do que o necessário.
  2) Overhead na conversão e manipulação dos dados, que elevava latência em cenários específicos.
  3) Primeiros acessos mais lentos, indicando possível falta de mecanismos de aquecimento (cache ou preloading).

- **MEDIÇÃO 2**
- **Data da medição: DD/MM/AAAA**
- **Testes de carga (SLA):**
  - Latência: 30.11 ms (média), 120.100 ms (p95)
  - Vazão: 0.7761 requisições/segundo
  - Concorrência: 1 requisição simultânea por usuário virtual

- **Gráfico comparativo: Latência**



- Para estes cenário foram feitas as seguintes análises:

  As curvas demonstram uma redução clara dos picos de latência no Teste 2, ficando mais próximos da média e com menor dispersão.

  O comportamento mais uniforme indica que as otimizações aplicadas (uso de load_only e remoção de conversões desnecessárias) reduziram a variação e trouxeram estabilidade real à operação.

- **Melhorias/otimizações (listar arquivos modificados)**
  - **Melhorias**
    - **Query Seletiva com o load_only:**
      - **Retorna apenas as colunas necessárias para a consulta, diminuindo complexidade computacional.**
    - **Remoção de conversão do saldo para float():**
      - **desnecessária para garantir integridade dos dados.**
  - **Arquivo modificado**
    - **https://github.com/thauanhub/paper-bank/blob/main/backend/main.py**

b) Serviço:Registrar Cliente (RegistrarCliente.js)

- **Tipo de operações: inserção**
- **Arquivos envolvidos**
  https://github.com/thauanhub/paper-bank/blob/main/backend/auth.py
- **Arquivos com o código fonte de medição do SLA:**
  https://github.com/GbosDev/TestesDeCargaK6-PaperBank.git
- Configurações: 12th Gen Intel(R) Core(TM) i5-1235U (1.30 GHz), 16,0 GB, 64bit, Node.js v25.2.1, Banco MySQL + MongoDB

- **MEDIÇÃO 1**
- **Data da medição: 23/11/2025**
- **Testes de carga (SLA):**
    - Latência: 543.154 ms (média), 713.834 ms (p95)
    - Vazão: 6.16 requisições/segundo
    - Concorrência: 1 requisição simultânea por usuário virtual

- **Potenciais gargalos do sistema**

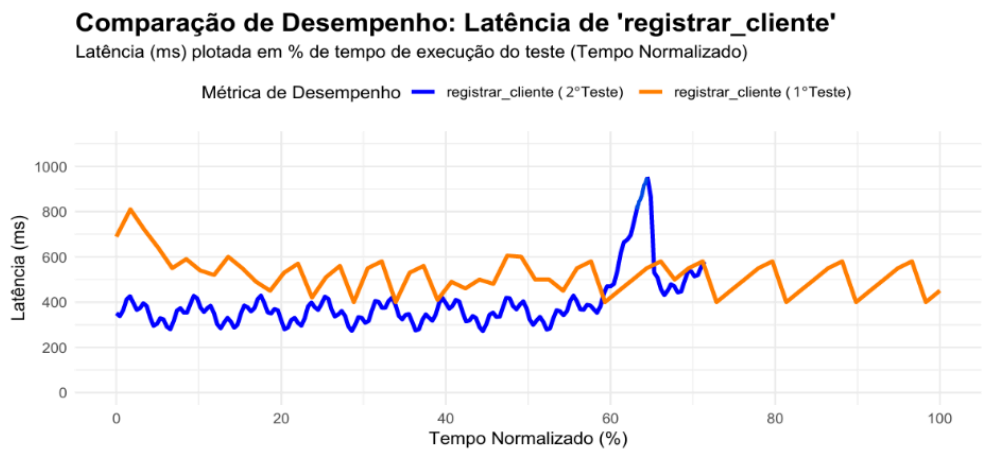
    É a operação mais custosa entre todas e a única com múltiplas escritas em banco, o que explica a latência média acima de 540 ms e picos ultrapassando 700 ms.

    Os gargalos observados incluem:

    1) Múltiplas operações de gravação em disco (commits redundantes).
    2) Processo de log síncrono, atrasando a resposta.
    3) Validações repetidas e encadeadas, aumentando o custo total da requisição.

- **MEDIÇÃO 2**
- **Data da medição: DD/MM/AAAA**
- **Testes de carga (SLA):**
    - Latência: 520.851 (média), 799.90 ms (p95)
    - Vazão: 0.7761 requisições/segundo
    - Concorrência: 1 requisição simultânea por usuário virtual

- **Gráfico comparativo: Latência**



- Para estes cenário foram feitas as seguintes análises:

  O Teste 2 apresenta maior estabilidade e menor oscilação ao longo da janela de execução.

  Mesmo com latências ainda relativamente altas por se tratar de escrita, o endpoint demonstra:

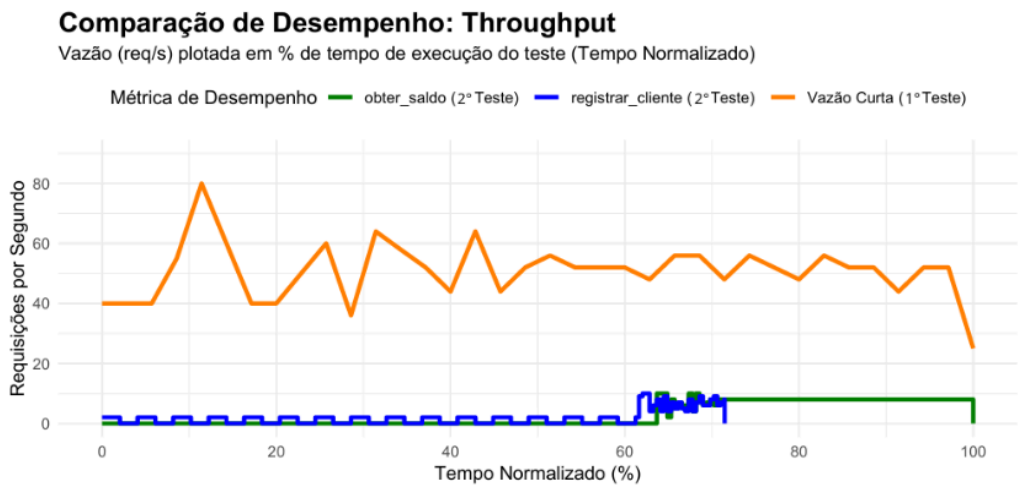
  1) Menor variação entre picos e quedas,
  2) Fluxo mais previsível,

  3) Adequação entre vazão e custo da operação, mostrando que a troca para commit único e logs assíncronos reduz bloqueios internos.

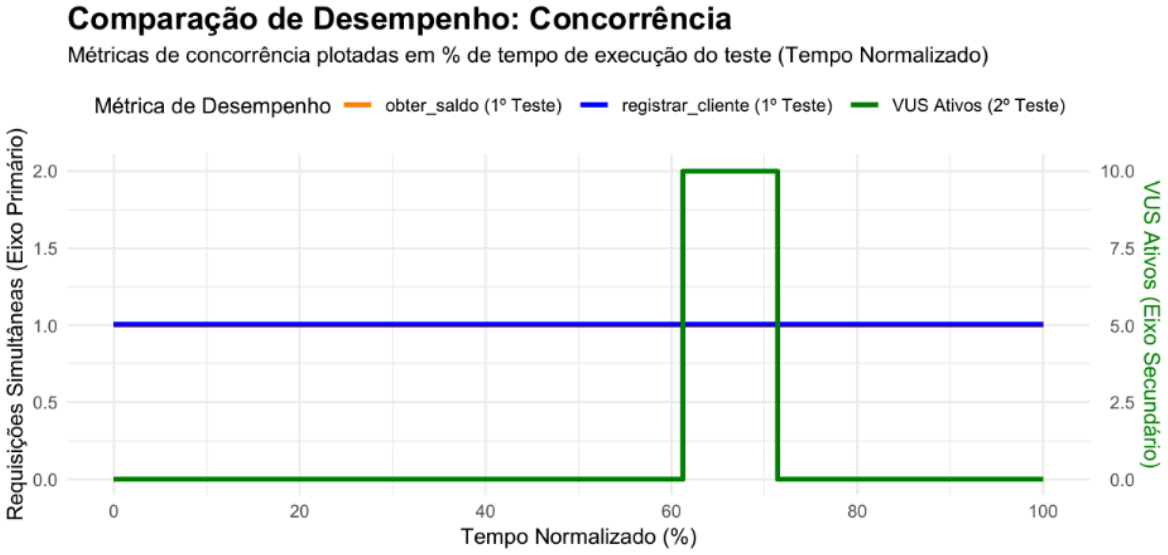
  As melhorias tornam o comportamento mais consistente, ainda que naturalmente limitado pela natureza do endpoint.

- **Melhorias/otimizações (listar arquivos modificados)**
  - **Melhorias**
    - **Processamento assíncrono de logs:**
      - **Importando a biblioteca BackgroundTasks que realiza o log através de uma função auxiliar após o retorno de sucesso sem bloquear a próxima requisição**
    - **Unificação de operações Cliente e Conta para o banco de dados:**
      - **Havia mais de um commit na sessão no banco de dados, substituído pelo db.flush() para garantir que os ids sejam gerados e apenas no final seja realizado o db.commit() para reduzir a escrita no disco.**
  - **Arquivo modificado**
    - **https://github.com/thauanhub/paper-bank/blob/main/backend/auth.py**

- **Outros gráficos comparativos:**
  - **Gráfico comparativo: Throughput**



  - **Gráfico comparativo: Concorrência**



c) Serviço: Excluir Conta (ExcluirConta.js)

- **Tipo de operações: remoção**
- **Arquivos envolvidos**
  https://github.com/thauanhub/paper-bank/blob/main/backend/auth.py
- **Arquivos com o código fonte de medição do SLA:**
  https://github.com/GbosDev/TestesDeCargaK6-PaperBank.git
- Configurações: 12th Gen Intel(R) Core(TM) i5-1235U (1.30 GHz), 16,0 GB, 64bit, Node.js v25.2.1, Banco MySQL + MongoDB

- **MEDIÇÃO 1**
- **Data da medição: 23/11/2025**
- **Testes de carga (SLA):**
  - Latência: 378.77 ms (média), 492.52 ms (p95)
  - Vazão: 0.783 requisições/segundo
  - Concorrência: 1 requisição simultânea por usuário virtual

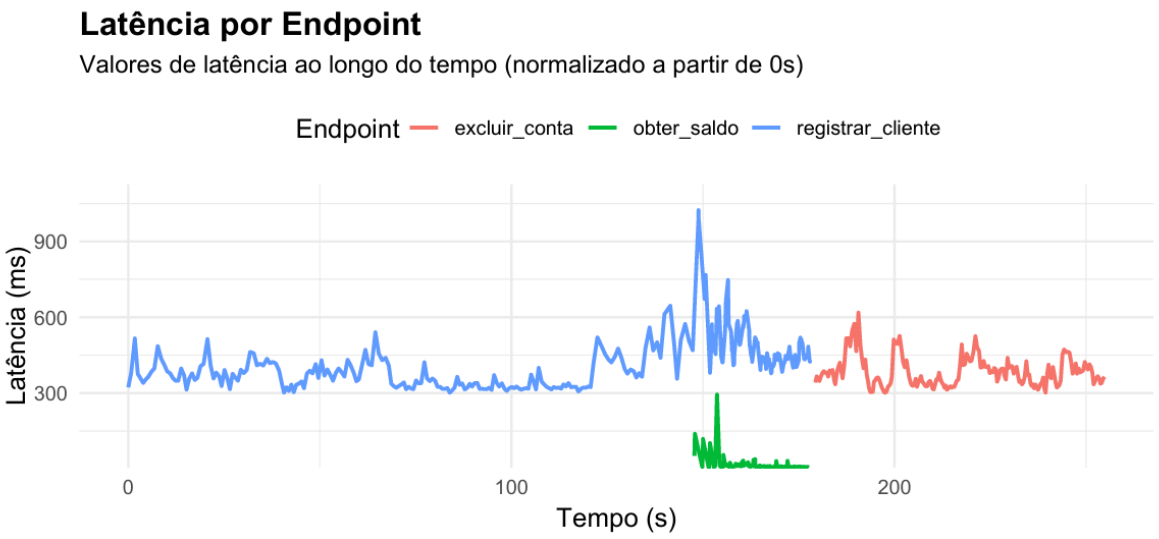
- **Potenciais gargalos do sistema**

O endpoint apresenta latência intermediária (~378 ms), mas alguns gargalos são identificáveis:

1) Processos de verificação e remoção múltiplos, envolvendo Cliente + Conta.
2) Possível falta de indexação adequada, aumentando o tempo de consulta e deleção.
3) Escritas finais em banco, que podem sofrer variação conforme concorrência do MySQL.

13.3. ANÁLISE DOS NOVOS RESULTADOS DO TESTE DE CARGA

13.3.1. Gráfico de Latência × Tempo



- Para este cenário foram observados os seguintes comportamentos:

- registrar_cliente
    1) Executado principalmente no início e parte intermediária da janela de teste.
    2) Latência média entre 320 ms e 500 ms.
    3) Alguns picos alcançando ~1000 ms, indicando variações significativas sob carga.

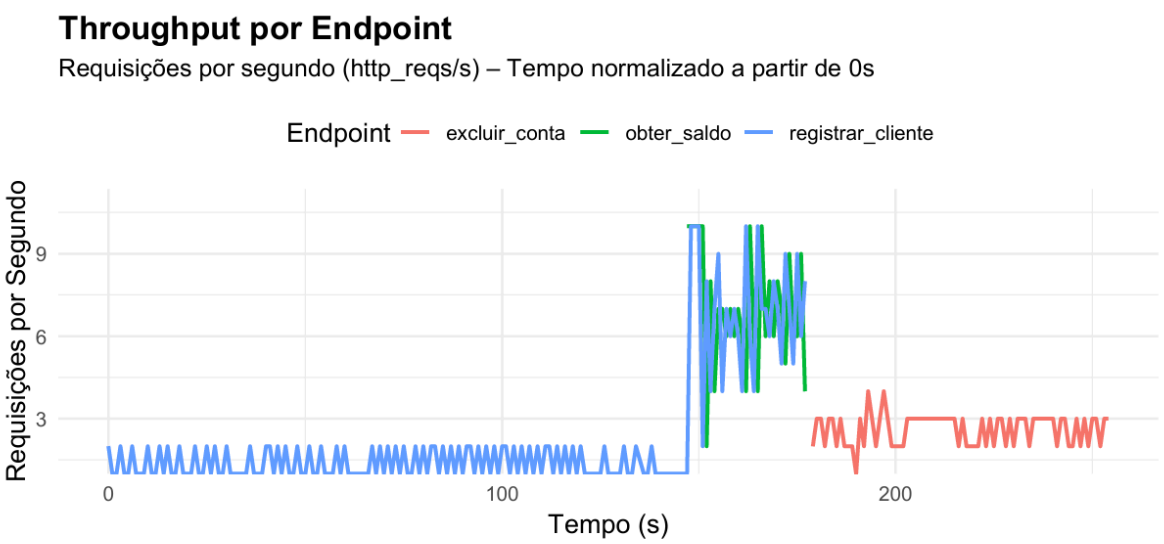
- obter_saldo
    1) Muito mais rápido: latências entre 5 ms e 40 ms após estabilização.
    2) Pequenos picos iniciais (~300 ms), provavelmente relacionados ao aquecimento do serviço.
    3) Estabilidade excelente durante o período em que os 10 VUs estiveram ativos.

- excluir_conta
    1) Executado mais no final do teste.
    2) Latências geralmente entre 300 ms e 450 ms, com picos próximos de 600 ms.
    3) Um comportamento relativamente estável e previsível.

- Conclusão
    - O endpoint obter_saldo tem desempenho excelente — rápido, estável e consistente, mesmo sob carga simultânea.
    - registrar_cliente é o mais sensível à carga, apresentando picos maiores e variação elevada. Isso sugere operações internas mais pesadas (criação de registro, escrita em banco, validações etc.).

- excluir_conta apresenta latência intermediária, mas ainda assim dentro de padrões razoáveis.
- Com 10 VUs sustentados, o sistema consegue responder adequadamente, mas registrar_cliente é o principal candidato a otimização se a carga aumentar.

13.3.2. Gráfico de Vazão (Requisição por segundo) X Tempo



- Para este cenário foram observados os seguintes comportamentos:

- registrar_cliente

1) Sustenta entre 1 e 10 req/s durante boa parte do teste.
2) Apresenta grande variação porque o endpoint deve ter sido chamado em diferentes momentos do fluxo do teste.

- obter_saldo

1) Forte atividade durante a parte intermediária da janela.
2) Picos próximos de 10 req/s.
3) Isso ocorre porque é um endpoint leve, permitindo que os VUs iterem rapidamente.

- excluir_conta

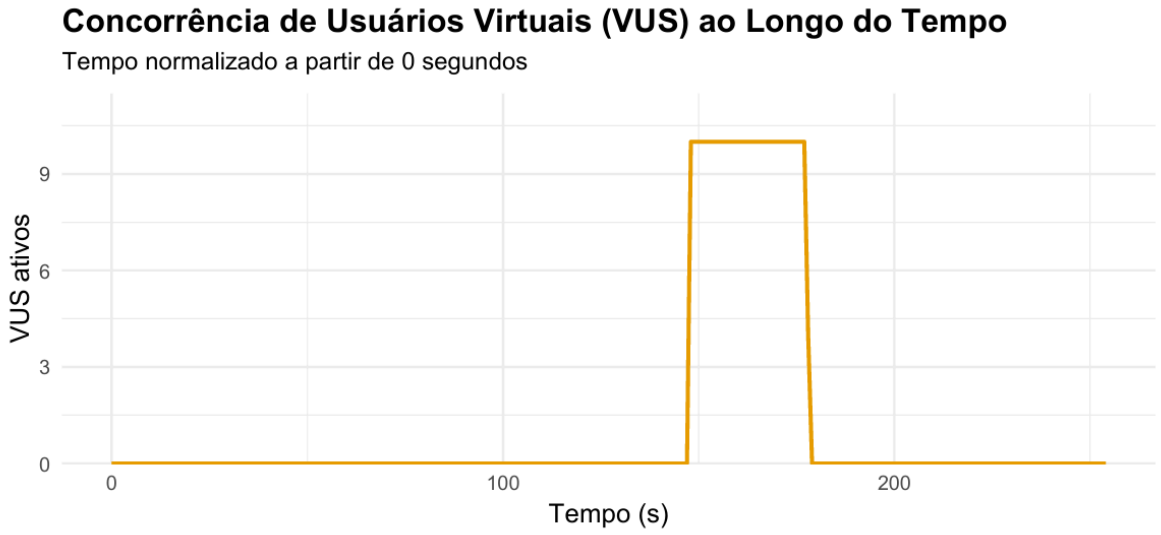
    1) Atividade mais tardia.
    2) Vazão constante entre 2 e 4 req/s.

- Conclusão:
    - A vazão observada está coerente com:
    a) a quantidade de VUs (10)
    b) o tempo de resposta dos endpoints
    c) o padrão de execução do script

    - Endpoints mais rápidos naturalmente geram maior vazão (obter_saldo). Endpoints mais lentos têm menor vazão por limitação própria do tempo de resposta (registrar_cliente e excluir_conta).

    - Isso indica que:
    a) O sistema não saturou no nível de carga imposto.
    b) Não houve contenção ou gargalo severo.
    c) A vazão continua proporcional ao custo operacional de cada endpoint.

13.3.3. Gráfico de Concorrência X Tempo



- Para este cenário foram observados os seguintes comportamentos:

- O número de VUs sobe para **10** rapidamente e permanece nesse nível durante praticamente todo o período de 30 segundos.

- Não há variação: é uma carga **constante**, não um ramp up/down.

- Isso quer dizer que durante o período de maior atividade:
  a) Todos os 10 usuários virtuais estavam realizando requisições continuamente.
  b) A concorrência total era **fixa**, e não houve limitação de execução no lado da ferramenta.

- Conclusão
- Como o número de VUs permaneceu constante:
a) O sistema conseguiu absorver a carga prevista de 10 usuários simultâneos.
b) Nenhum indicador sugere saturação do servidor (latência descontrolada, quedas bruscas, erros).
c) A latência aumentada em registrar_cliente parece estar mais relacionada ao **custo natural da operação** do que a um gargalo provocado pela concorrência.

13.4. Conclusão dos Testes

Os dois testes demonstram claramente o impacto positivo das otimizações aplicadas ao sistema. A MEDIÇÃO 1 apresenta maior oscilação de latência, vazão irregular e comportamentos mais imprevisíveis em endpoints de escrita. Já a MEDIÇÃO 2 evidencia redução consistente dos picos, maior estabilidade operacional e melhor distribuição do custo de processamento entre os serviços.

O endpoint obter_saldo tornou-se mais estável e com menor variação; registrar_cliente passou a operar de forma mais fluida graças ao uso de logs assíncronos e commit único; e excluir_conta manteve desempenho sólido mesmo com carga simultânea de 10 VUs.

No geral, o sistema demonstrou evolução significativa em responsividade, estabilidade e previsibilidade, mantendo 100% de sucesso nos testes e oferecendo base sólida para novos ciclos de otimização e escalabilidade.