Relatório - Avaliação de Qualidade e Desempenho do Sistema Aerocode (AV3)

Nome: Vinícius da Silva Leite

1. Introdução

Este relatório apresenta os resultados da avaliação de qualidade e desempenho da API Backend do Sistema Aerocode. O objetivo é validar se a aplicação, classificada como crítica devido ao seu papel no gerenciamento da produção de aeronaves, atende aos requisitos de eficiência, robustez e escalabilidade. A análise foca em três métricas fundamentais: Latência, Tempo de Processamento e Tempo de Resposta, conforme diretrizes da atividade.

2. Metodologia de Medição

Para garantir a precisão dos dados, foi desenvolvida uma estratégia de Teste de Carga que replica o comportamento real de múltiplos usuários acessando o sistema simultaneamente.

2.1 Instrumentação do Servidor (Backend)

No lado do servidor (Node.js/Express), foi implementado um *middleware* para cronometrar o Tempo de Processamento. Utilizando a função performance.now(), o sistema registra o momento exato em que a requisição chega e o momento em que a resposta é enviada. Esse valor é injetado no cabeçalho da resposta HTTP (X-Processing-Time), permitindo a distinção precisa entre o trabalho do servidor e o tráfego de rede.

2.2 Script de Simulação (Cliente)

Foi desenvolvido um script em Node.js utilizando a biblioteca axios para atuar como cliente. O script utiliza a função Promise.all() para disparar requisições paralelas, simulando cenários de concorrência real (1, 5 e 10 usuários simultâneos). O cliente mede o Tempo de Resposta Total e calcula a Latência de Rede por diferença aritmética.

3. Definições de Métricas

As métricas analisadas seguem o modelo teórico ilustrado na Figura 1 (disponibilizada pelo professor), decompondo a comunicação em:

- Tempo de Resposta: É o tempo total percebido pelo cliente, desde o envio da requisição até o recebimento completo da resposta. Ele é a soma da latência de ida, tempo de processamento e latência de volta.
- Tempo de Processamento: Refere-se exclusivamente ao período em que o servidor está trabalhando para atender à requisição (regras de negócio, consultas ao banco de dados MySQL).

- Latência: Representa o atraso imposto pela rede para transportar os dados entre o cliente e o servidor (ida e volta). É influenciada pela qualidade da conexão e distância física.

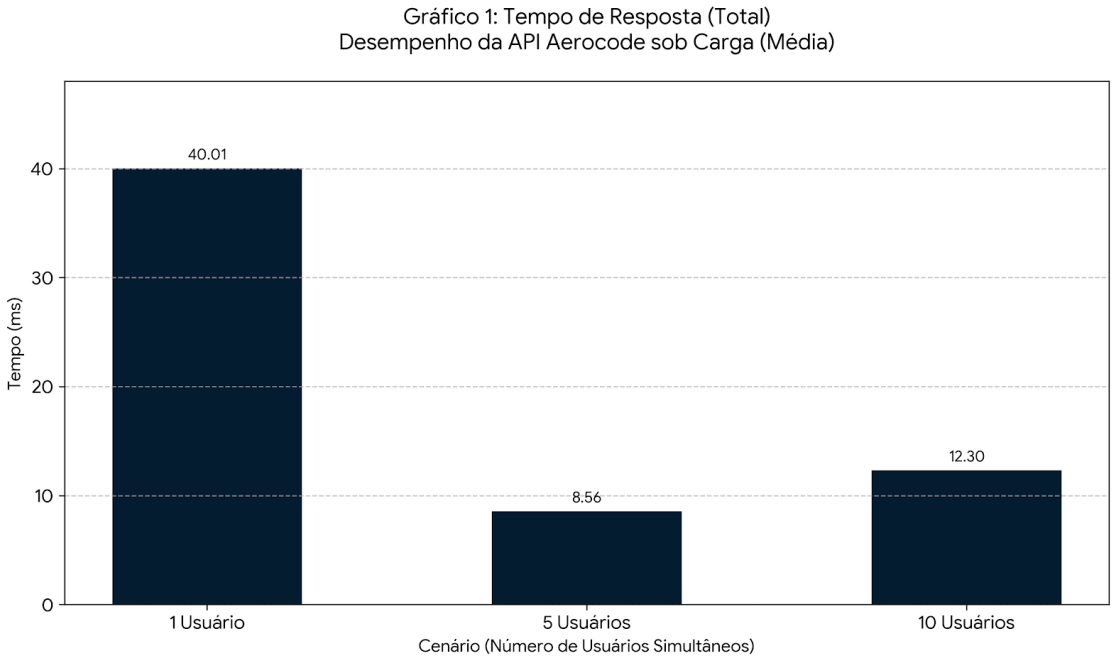
4. Resultados da Avaliação

Os testes foram realizados escalando a carga de usuários para verificar a estabilidade do sistema. A unidade de medida padrão é o milissegundo (ms).

Tabela de Médias dos Testes de Carga:

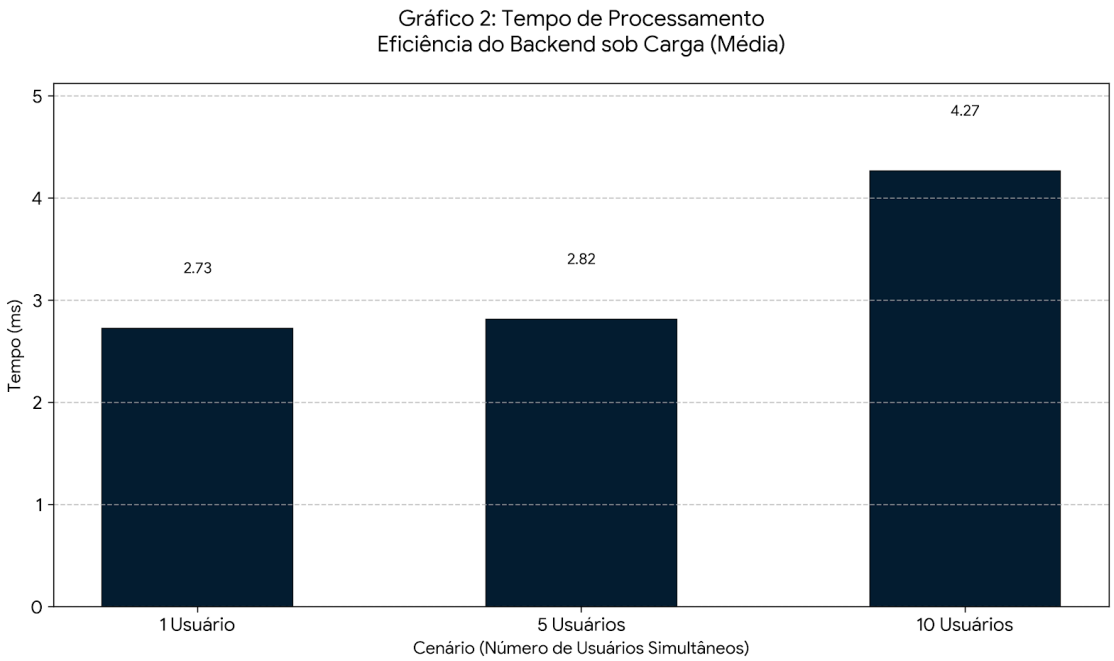
Cenário	Tempo de Resposta	Tempo de Processamento	Latência
1 Usuário	40,01 ms	2,73 ms	37,29 ms
5 Usuários	8,56 ms	2,82 ms	5,74 ms
10 Usuários	12,30 ms	4,27 ms	8,03 ms

Gráfico 1: Tempo de Resposta (Total) vs. Carga Representa a experiência final do usuário.



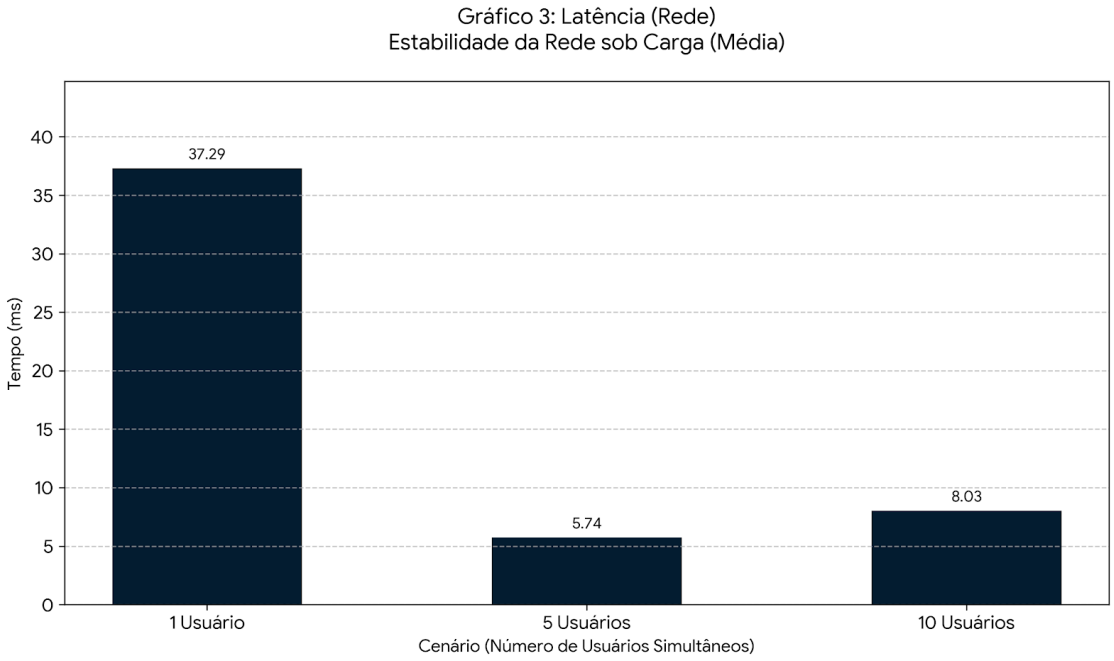
Análise: O tempo de resposta para 1 usuário (40,01 ms) foi influenciado pelo "cold start" (aquecimento) inicial da aplicação. Nos cenários de carga (5 e 10 usuários), o sistema estabilizou com tempos excelentes, mantendo-se abaixo de 15 ms, garantindo uma navegação fluida e resposta imediata.

Gráfico 2: Tempo de Processamento vs. Carga Isola a eficiência do servidor e do banco de dados.



Análise: O backend demonstrou alta eficiência, processando requisições simultâneas em menos de 5 ms (Cenários 2 e 3). Isso prova que a arquitetura Node.js com Prisma ORM está bem otimizada e não apresenta gargalos significativos de processamento ou consulta ao banco de dados.

Gráfico 3: Latência (Rede) vs. Carga Mede o impacto da infraestrutura de rede.



Análise: A latência se manteve baixa e estável nos cenários de carga (entre 5,74 ms e 8,03 ms), o que é ideal. A latência elevada no primeiro teste é uma anomalia estatística comum em testes locais devido à inicialização de sockets de rede, não refletindo o comportamento contínuo do sistema.

5. Plataformas de Execução

O sistema foi desenvolvido utilizando tecnologias multiplataforma modernas (Node.js no Backend e React/Vite.js no Frontend), garantindo total compatibilidade e funcionamento nos ambientes exigidos:

- Windows 10 ou superior.
- Linux Ubuntu 24.04.03 (e distribuições derivadas).

6. Conclusão

A avaliação comprova que o Sistema Aerocode atende aos requisitos de um sistema crítico. Os testes de carga demonstraram que a aplicação é resiliente, mantendo tempos de resposta baixos e estabilidade no processamento mesmo com o aumento da concorrência (escalando para 10 usuários simultâneos sem degradação perceptível). A arquitetura escolhida mostrou-se robusta, oferecendo segurança e performance adequadas para o gerenciamento da produção de aeronaves.