Rafael Parreira Chequer
rafaelpchequer@gmail.com

Documento de Visão para o Sistema Stream Sentry

28 de Setembro de 2025

Proposta do aluno Rafael Parreira Chequer ao curso de Engenharia de Software como projeto de Trabalho de Conclusão de Curso (TCC), tendo como orientadores os professores: Cleiton Silva Tavares, Danilo de Quadros Maia Filho, Leonardo Vilela Cardoso e Raphael Ramos Dias Costa.

INTRODUÇÃO

Este Documento de Visão apresenta o sistema *Stream Sentry*, uma ferramenta web desenvolvida em React para automação de testes *end-to-end* em aplicações de videoconferência baseadas em *WebRTC* ou *APIs* de videoconferência, como Zoom SDK e Jitsi. O documento define o escopo do sistema, identifica os *stakeholders*, levanta suas necessidades e descreve as funcionalidades a serem implementadas, atendendo aos requisitos do Trabalho de Conclusão de Curso (TCC) do curso de Engenharia de Software da PUC Minas.

OBJETIVOS

Este documento apresenta a visão do sistema **Stream Sentry**, uma ferramenta web desenvolvida em React para automação de testes *end-to-end* em aplicações de videoconferência baseadas em *WebRTC* ou APIs de videoconferência (ex.: Zoom SDK, Jitsi). O objetivo é definir o escopo do sistema, identificar itens fora do escopo, mapear gestores, usuários e outros interessados, levantar suas necessidades e descrever as funcionalidades a serem implementadas. O sistema visa atender às demandas de desenvolvedores, engenheiros de *Quality Assurance* (QA), equipes ágeis e pesquisadores, promovendo maior eficiência no ciclo de desenvolvimento de software por meio da automação de testes, redução de bugs em produção e geração de relatórios detalhados.

ESCOPO

O **Stream Sentry** é uma ferramenta web *open-source* (licença MIT) que automatiza testes *end-to-end* para aplicações de videoconferência desenvolvidas em React. Suas principais funções incluem:

- **Simulação de cenários reais**: Testa conexões com múltiplos usuários virtuais, verificando estabilidade, qualidade de áudio/vídeo e tratamento de erros.
- **Execução de testes automatizados**: Utiliza *headless browsers* (ex.: Puppeteer) para simular interações em navegadores reais.
- **Geração de relatórios**: Produz métricas detalhadas, como tempo de conexão, falhas detectadas e cobertura de testes, exportáveis em formatos como JSON ou CSV.
- **Compatibilidade**: Suporta aplicações baseadas em *WebRTC* ou APIs de videoconferência, utilizando JavaScript/TypeScript.

O sistema é voltado para desenvolvedores React, engenheiros de QA, equipes ágeis e pesquisadores acadêmicos que estudam automação de testes multimídia. Ele substituirá processos manuais de validação, que são demorados e suscetíveis a erros, mas não substituirá sistemas específicos existentes, pois é uma ferramenta nova no contexto da organização.

Sistemas atuais/concorrentes:

- **Sistemas atuais na organização**: Não há sistemas específicos para automação de testes *end-to-end* de videoconferências na organização conhecidos pelo autor. Testes manuais são predominantes, realizados por equipes de QA com ferramentas genéricas como Selenium ou Cypress, que não são otimizadas para *WebRTC*.
- **Sistemas concorrentes**: Ferramentas como TestRTC e BrowserStack oferecem testes para *WebRTC*, mas são pagas, complexas ou focadas em cenários específicos, sem a flexibilidade *open-source* do StreamSentry.

FORA DO ESCOPO

Os seguintes itens foram analisados, mas excluídos do escopo do **Stream Sentry**:

- Suporte a sistemas legados baseados em tecnologias obsoletas (ex.: Flash).
- Testes de desempenho em larga escala com milhares de usuários simultâneos, devido a limitações de infraestrutura.
- Integração direta com pipelines de Integração Contínua e Entrega Contínua (CI/CD), embora os relatórios sejam compatíveis com essas ferramentas.
- Testes de segurança (ex.: verificação de vulnerabilidades como injeção de código), que não é o foco principal.

- Desenvolvimento de uma interface mobile nativa, pois o sistema será acessado via navegadores web em desktops.

GESTORES, USUÁRIOS E OUTROS INTERESSADOS

Nome	Rafael Parreira Chequer
Qualificação	Estudante
Responsabilidades	Desenvolvedor do projeto, responsável por planejar, implementar e documentar o Stream Sentry, garantindo alinhamento com a Resolução de TCC I.

LEVANTAMENTO DE NECESSIDADES

1. **Automação de testes *end-to-end***
   Justificativa: Desenvolvedores React e equipes ágeis precisam reduzir o tempo e o esforço gastos em testes manuais de videoconferências, que são propensos a erros e ineficientes. A automação aumenta a cobertura de testes e a confiabilidade das aplicações.
2. **Geração de relatórios detalhados**
   Justificativa: Engenheiros de QA necessitam de métricas claras (ex.: latência, falhas, cobertura) para identificar e corrigir problemas rapidamente, melhorando a qualidade do software.
3. **Flexibilidade e extensibilidade *open-source***
   Justificativa: Pesquisadores acadêmicos requerem uma ferramenta acessível e modificável para experimentação em automação de testes multimídia, enquanto equipes ágeis buscam integração com seus fluxos de trabalho.
4. **Conformidade com requisitos acadêmicos**
   Justificativa: Professores de TCC I exigem que o projeto atenda à Resolução de TCC I, com documentação clara e foco em automação e eficiência.

FUNCIONALIDADES DO PRODUTO

Necessidade: Automação de testes *end-to-end*

Funcionalidade	**Categoria**
1. Simulação de múltiplos usuários	Crítico
2. Testes de estabilidade e qualidade (latência, áudio/vídeo)	Crítico

Necessidade: Geração de relatórios detalhados	
Funcionalidade	**Categoria**
1. Geração de relatórios com métricas (JSON/CSV)	Crítico

Necessidade: Flexibilidade e extensibilidade *open-source*	
Funcionalidade	**Categoria**
1. Compatibilidade com *WebRTC*/APIs de videoconferência	Importante
2. Código *open-source* sob licença MIT	Importante

Necessidade: Conformidade com requisitos acadêmicos	
Funcionalidade	**Categoria**
1. Interface web em React para configuração e visualização	Importante
2. Documentação completa (Documento de Visão, Manual do Usuário, Documentação Técnica)	Crítico

INTERLIGAÇÃO COM OUTROS SISTEMAS

O **VideoTestHub** será uma ferramenta standalone, mas poderá interagir com:

- ***Headless browsers***: Utilizará Puppeteer para simular interações do usuário em navegadores reais (Chrome, Firefox, Edge).
- **APIs de videoconferência**: Compatível com APIs como Zoom SDK ou Jitsi para testar aplicações específicas.
- **Sistemas de CI/CD**: Gera relatórios exportáveis (JSON/CSV) que podem ser processados por ferramentas como Jenkins ou GitHub Actions, embora não haja integração direta.

RESTRIÇÕES

- **Compatibilidade**: Funcionar em navegadores modernos (Chrome, Firefox, Edge) com suporte a *WebRTC*.
- **Desempenho**: Suportar testes com até 10 usuários simultâneos em hardware padrão (ex.: 8 GB RAM, processador i5).
- **Licenciamento**: Ser *open-source* sob licença MIT, garantindo reutilização.
- **Acessibilidade**: Interface web acessível em desktops, sem dependências proprietárias.
- **Escalabilidade**: Não suportar testes com milhares de usuários devido a limitações de infraestrutura.
- **Restrições legais**: Nenhuma, desde que as APIs utilizadas (ex.: Zoom SDK) sejam acessadas conforme suas licenças.

DOCUMENTAÇÃO

Os seguintes documentos serão fornecidos:

- **Documento de Visão**: Este documento, detalhando escopo, usuários, necessidades e funcionalidades, em formato *.md*.
- **Manual do Usuário**: Guia em *.md* para configurar, executar testes e interpretar relatórios.
- **Documentação Técnica**: Descrição da arquitetura, tecnologias (React, Node.js, Puppeteer) e instruções de instalação, em *.md*.
- **Código-fonte**: Disponibilizado em repositório público (ex.: GitHub) sob licença MIT.