

---

# Documentação de Projeto

para o sistema

## StreamSentry

**Versão 1.0**

Projeto de sistema elaborado pelo aluno Rafael Parreira Chequer e apresentado ao curso de Engenharia de Software da PUC Minas como parte do Trabalho de Conclusão de Curso (TCC) sob orientação de conteúdo do professor a ser definido, orientação acadêmica do professor a ser definido e orientação de TCC II do professor (a ser definido no próximo semestre).

**17 de setembro de 2025**

# Tabela de Conteúdo

<b>Tabela de Conteúdo</b>	<b>ii</b>
<b>Histórico de Revisões</b>	<b>ii</b>
<b>1. Introdução</b>	<b>1</b>
<b>2. Modelo de Usuários e Requisitos</b>	<b>1</b>
2.1 Descrição de Atores	1
2.2 Modelos de Usuário	1
2.3 Modelo de Casos de Uso e Histórias de Usuários	1
2.4 Diagrama de Sequência do Sistema e Contrato de Operações	1
2.5 Diagramas de Estados	1
2.6 Diagrama de Componentes	1
<b>3. Projeto de Interface com Usuário</b>	<b>2</b>
3.1 Interfaces Comuns a Todos os Atores	2
3.2 Interfaces Usadas pelo Ator <A>	2
3.3 Interfaces Usadas pelo Ator <B>	2
<b>4. Modelos de Projeto</b>	<b>2</b>
<b>5. Projeto de Interface com Usuário</b>	<b>2</b>
<b>6. Glossário e Modelos de Dados</b>	<b>2</b>
<b>7. Casos de Teste</b>	<b>2</b>
<b>8. Cronograma e Processo de Implementação</b>	<b>2</b>

## Histórico de Revisões

Nome	Data	Razões para Mudança	Versão
Rafael Parreira Chequer	17/09/2025	Criação inicial do documento para entrega da A3	1.0

# 1. Introdução

Este documento apresenta a análise de usuário, contexto, interface e interação do sistema **StreamSentry**, uma ferramenta web *open-source* (licença MIT) 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). A referência principal para a descrição do problema, domínio e requisitos é o **Documento de Visão** (anexo a este documento), que detalha o escopo, as funcionalidades, os usuários-alvo (desenvolvedores React, engenheiros de QA (Quality Assurance), equipes ágeis e pesquisadores acadêmicos) e as restrições do sistema. Este documento atende às exigências da Resolução de TCC I, incluindo a elaboração de modelos de usuário, diagrama de casos de uso, histórias de usuário e projeto de interfaces, conforme as boas práticas de engenharia de software.

O objetivo desta entrega é prover: (1) o diagrama de casos de uso e histórias de usuário, (2) modelos de usuários na forma de personas e (3) wireframes de baixa fidelidade para as interfaces do sistema, todos devidamente contextualizados. Os artefatos estão armazenados na pasta **Artefatos** do repositório GitHub Classroom, conforme as instruções da atividade.

## 2. Modelos de Usuário e Requisitos

### 2.1 Descrição de Atores

Os atores que interagem com o sistema **StreamSentry** são definidos com base nos usuários-alvo descritos no **Documento de Visão**. Abaixo está a descrição de cada ator:

- **Desenvolvedor React:** Profissional que desenvolve aplicações de videoconferência em React, utilizando *WebRTC* ou *APIs* como Zoom SDK ou Jitsi. Este ator utiliza o **Stream Sentry** para automatizar testes *end-to-end*, reduzindo o tempo e o esforço em validações manuais.
- **Engenheiro de QA:** Responsável por garantir a qualidade de aplicações de videoconferência, utilizando o sistema para executar testes automatizados e analisar relatórios detalhados com métricas como latência, falhas e cobertura de testes.
- **Pesquisador Acadêmico:** Estudante ou professor que investiga automação de testes multimídia, utilizando o **Stream Sentry** como uma ferramenta *open-source* para experimentação e análise.
- **Equipe Ágil:** Grupo de profissionais (desenvolvedores, testadores e gerentes) que utilizam o sistema em fluxos de trabalho ágeis para integrar testes automatizados e relatórios em seus processos.

## 2.2 Modelos de Usuários

Para representar os usuários do sistema, foram desenvolvidas **personas** que refletem as características, necessidades e objetivos dos atores identificados. As personas são baseadas nas necessidades levantadas no **Documento de Visão** e ajudam a guiar o design do sistema.

### Persona 1:



**Lucas**

**Desenvolvedor  
React**

- **Idade:** 28 anos
- **Formação:** Bacharel em Ciência da Computação
- **Contexto:** Trabalha em startup desenvolvendo aplicações WebRTC
- **Objetivos:** Automatizar testes end-to-end para estabilidade e redução de bugs
- **Necessidades:** Interface simples, relatórios JSON/CSV, suporte a WebRTC
- **Frustrações:** Testes manuais demorados, ferramentas pagas caras
- **Cenário de Uso:** Configura teste com 5 usuários virtuais no StreamSentry e exporta relatório JSON

### Persona 2:



**Ana**

**Engenheira de  
QA**

- **Idade:** 32 anos
- **Formação:** Engenharia de Software
- **Contexto:** Valida qualidade de áudio/vídeo em APIs como Jitsi
- **Objetivos:** Executar testes automatizados com métricas de latência e falhas
- **Necessidades:** Relatórios detalhados, interface intuitiva, suporte a headless browsers
- **Frustrações:** Ferramentas como Selenium não otimizadas para WebRTC
- **Cenário de Uso:** Configura teste de estabilidade com 10 usuários e exporta relatório CSV

**Persona 3:**



**Carla**

**Pesquisadora Acadêmica**

- **Idade:** 40 anos
- **Formação:** Doutora em Ciência da Computação
- **Contexto:** Pesquisa testes multimídia com ferramentas open-source
- **Objetivos:** Testar hipóteses com ferramenta flexível
- **Necessidades:** Código open-source (MIT), documentação, suporte a Zoom SDK
- **Frustrações:** Ferramentas proprietárias e falta de documentação
- **Cenário de Uso:** Configura teste com Jitsi, modifica código e documenta resultados

## 2.3 Modelo de Casos de Uso e Histórias de Usuários

O diagrama de casos de uso do sistema **StreamSentry** foi desenvolvido para representar as interações principais dos atores com o sistema, conforme as funcionalidades descritas no **Documento de Visão**. O diagrama está disponível na pasta **Artefatos** do repositório com o nome **UseCaseDiagram.puml** e é apresentado na Figura 1.

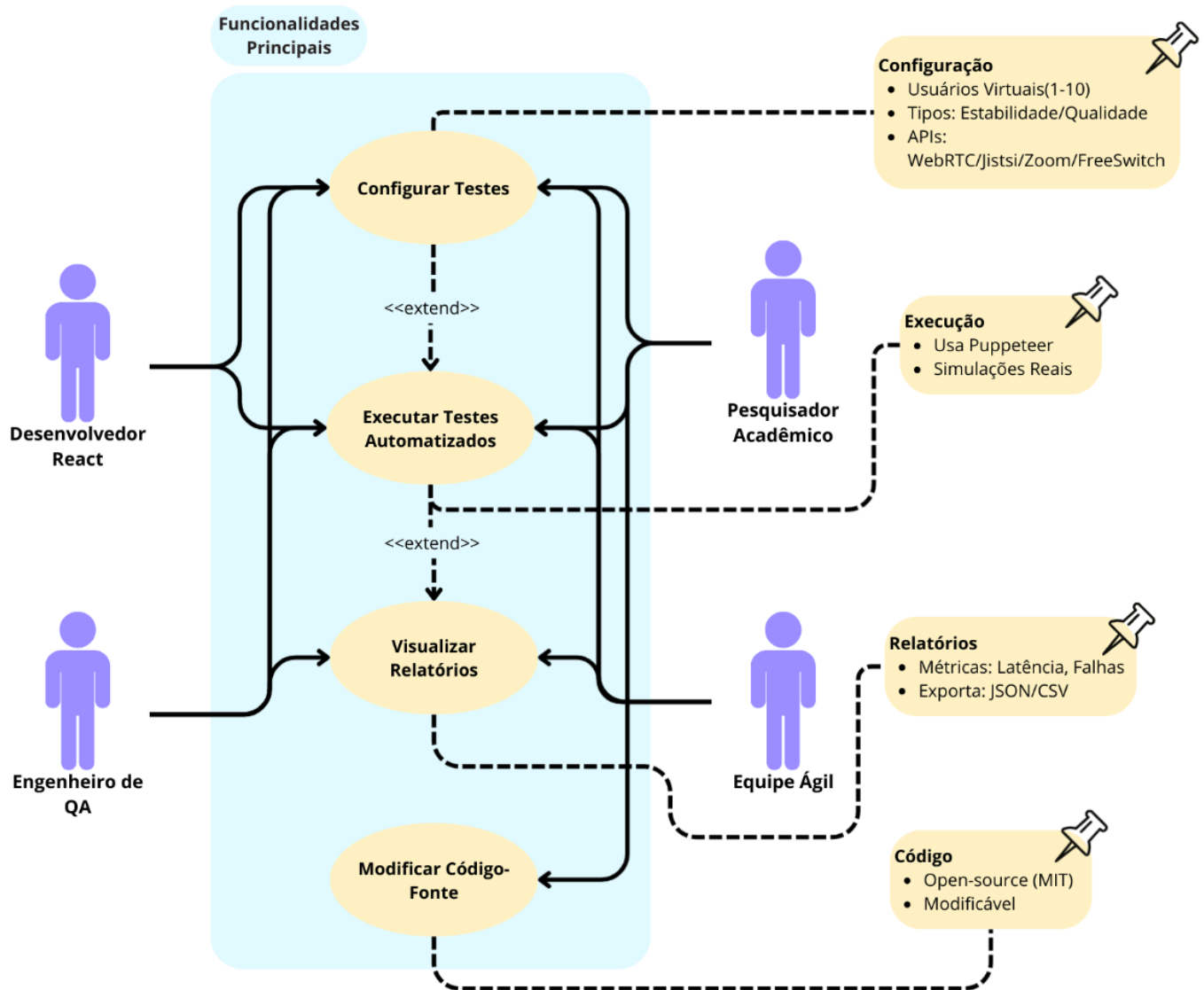


Figura 1: Diagrama de Casos de Uso do StreamSentry

O diagrama ilustra os atores (Desenvolvedor React, Engenheiro de QA, Pesquisador Acadêmico, Equipe Ágil) e suas interações com os casos de uso principais: configurar testes, executar testes

automatizados, visualizar relatórios e modificar código-fonte. Cada caso de uso reflete uma funcionalidade crítica ou importante identificada no Documento de Visão.

Abaixo está a lista de histórias de usuário, mapeadas com base nas necessidades e funcionalidades do sistema, utilizando identificadores únicos para referência:

- **US01:** Como Desenvolvedor React, eu quero configurar um teste com múltiplos usuários virtuais, para verificar a estabilidade da conexão em uma aplicação *WebRTC*, garantindo que a funcionalidade seja testada rapidamente.
- **US02:** Como Engenheiro de QA, eu quero executar testes automatizados de qualidade de áudio e vídeo, para identificar falhas de latência ou interrupções antes da entrega do software.
- **US03:** Como Engenheiro de QA, eu quero visualizar relatórios detalhados em formato JSON ou CSV, para compartilhar métricas de teste com a equipe e tomar decisões baseadas em dados.
- **US04:** Como Pesquisador Acadêmico, eu quero acessar o código-fonte do StreamSentry, para modificá-lo e experimentar novos cenários de teste multimídia, atendendo aos meus objetivos de pesquisa.
- **US05:** Como membro de uma Equipe Ágil, eu quero configurar testes via uma interface web intuitiva, para integrar a automação ao fluxo de trabalho do time sem complicações.
- **US06:** Como Desenvolvedor React, eu quero que o sistema suporte *APIs* de videoconferência como Jitsi, para testar aplicações específicas sem necessidade de ferramentas adicionais.
- **US07:** Como Pesquisador Acadêmico, eu quero acessar documentação técnica completa, para entender a arquitetura do sistema e facilitar sua modificação para experimentos acadêmicos.

## 2.4 Diagrama de Sequência do Sistema e Contrato de Operações

Nesta subseção é apresentado o diagrama de sequência do sistema e os Contratos de Operações.

Formato para cada contrato de operação

<b>Contrato</b>	
<b>Operação</b>	
<b>Referências cruzadas</b>	
<b>Pré-condições</b>	
<b>Pós-condições</b>	

## 3. Projeto de Interface com Usuário

### 3.1 Interfaces Comuns a Todos os Atores

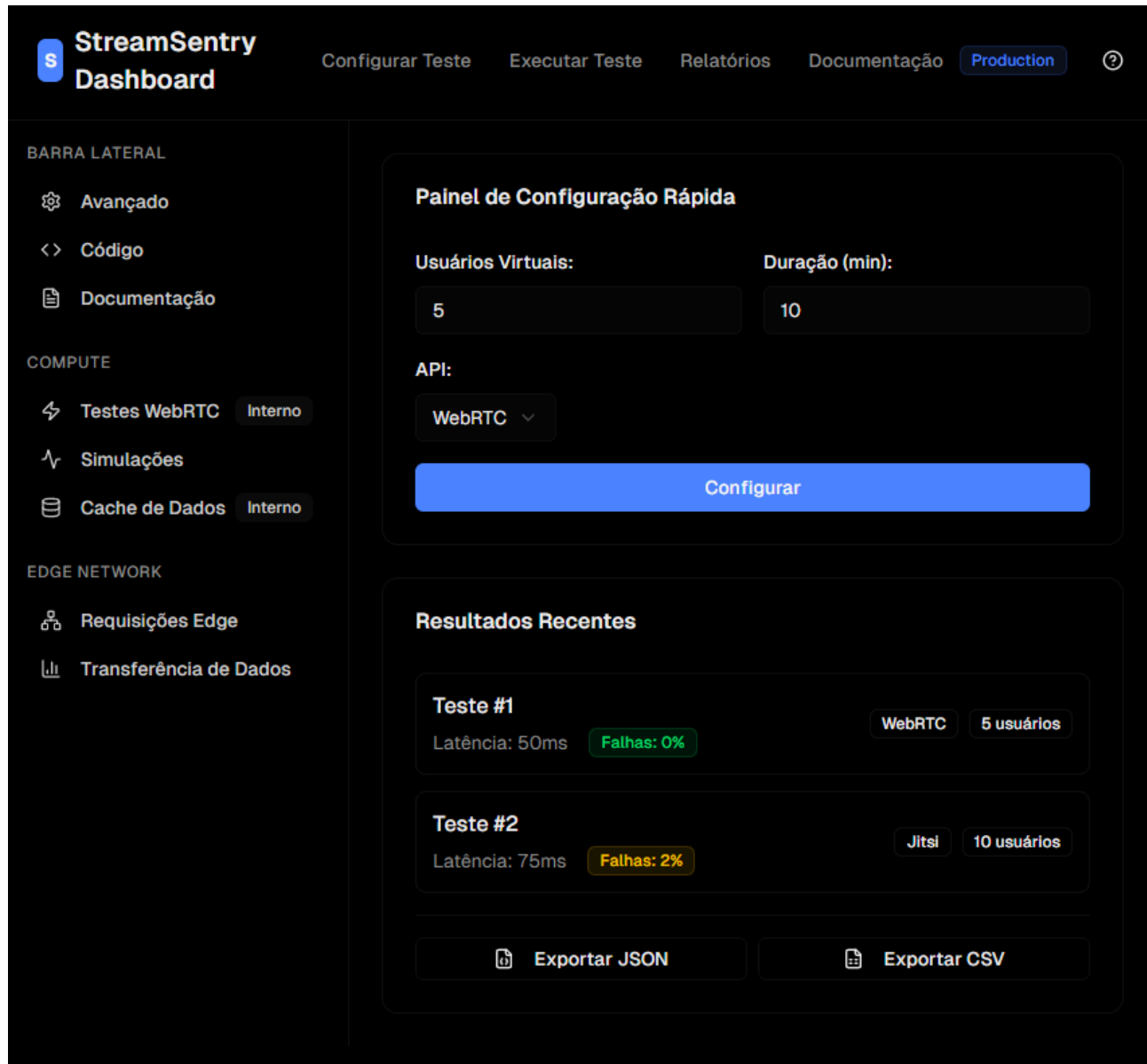
O *Dashboard* do sistema StreamSentry é a interface principal, acessível a todos os atores: Desenvolvedor React, Engenheiro de QA, Pesquisador Acadêmico e Equipe Ágil. Ele serve como ponto central para gerenciar testes automatizados de videoconferência, com base nas personas descritas na seção 2.2. O wireframe está disponível na pasta *Artefatos* do repositório, nomeado *DashboardWireframe.png*.

A interface, projetada para ser intuitiva e minimalista, inclui um menu superior com opções como "Configurar Teste", "Executar Teste", "Relatórios" e "Documentação" (exclusiva para pesquisadores). O painel de configuração rápida permite definir parâmetros como número de usuários virtuais, tipo de API (*WebRTC*, Jitsi, Zoom SDK) e duração do teste. A área de resultados exibe métricas como latência média, taxa de falhas e cobertura de teste.

Um botão de exportação suporta relatórios em JSON ou CSV, enquanto a barra lateral oferece links para configurações avançadas e código-fonte. Lucas, o Desenvolvedor React, configura testes com 5 usuários virtuais para verificar a estabilidade. Ana, a Engenheira de QA, executa testes de áudio/vídeo e exporta relatórios em CSV.

Dr. Carla, a Pesquisadora Acadêmica, acessa documentação e código-fonte para experimentos. A Equipe Ágil integra testes ao fluxo de trabalho com configurações simplificadas. O design minimalista garante usabilidade para usuários técnicos e agilidade para equipes, atendendo às demandas de Ana e Dr. Carla.





**Figura 2: Wireframe do *Dashboard* do StreamSentry**

(A imagem está disponível na pasta Artefatos do repositório, sob o nome *DashboardWireframe.png*)

O Dashboard equilibra simplicidade e funcionalidade, assegurando flexibilidade e conformidade com os objetivos do sistema.

## 4. Modelos de Projeto

### 4.1 Diagrama de Classes

Diagrama de classes do sistema

## **4.2 Diagramas de Sequência**

Diagramas de sequência para realização de casos de uso.

## **4.3 Diagramas de Comunicação**

Diagramas de comunicação para realização de casos de uso.

## **4.4 Arquitetura**

Pode ser descrita com um diagrama apropriado da UML ou C4 Model

## **4.5 Diagramas de Estados**

Diagramas de estados do sistema.

## **4.6 Diagrama de Componentes e Implantação.**

Diagramas de componentes do sistema. Diagrama de implantação mostrando onde os componentes estarão alocados para a execução.

# **5. Projeto de Interface com Usuário**

## **5.1 Esboço das Interfaces Comuns a Todos os Atores**

Wireframe/mockup/storyboard das interfaces que são comuns a todos os atores do sistema.

## **5.2 Esboço das Interfaces Usadas pelo Ator <A>**

Wireframe/mockup/storyboard das interfaces exclusivas do ator <A>

## **5.3 Esboço das Interfaces Usadas pelo Ator <B>**

Wireframe/mockup/storyboard das interfaces exclusivas do ator <B>

## **6. Glossário e Modelos de Dados**

Deve-se apresentar o glossário para o sistema. Também apresente esquemas de banco de dados e as estratégias de mapeamento entre as representações de objetos e não-objetos.

## **7. Casos de Teste**

Uma descrição de casos de teste para validação do sistema.

## **8. Cronograma e Processo de Implementação**

Uma descrição do cronograma para implementação do sistema e do processo que será seguido durante a implementação.