

# Plano de Teste: Sistema para Controle de Frequência

## IDENTIFICAÇÃO DO PROJETO

### Projeto

- Registra

### Orientador

- Alysson Filgueira Milanez

## HISTÓRICO DE REGISTROS

Versão	Data	Autor	Descrição
1.0	14/05/2025	Iverton Emiquison R. Bessa Matheus Marques Nunes	Plano de teste inicial

---

## SUMÁRIO

<b>1. INTRODUÇÃO.....</b>	<b>3</b>
<b>2. RESUMO.....</b>	<b>3</b>
<b>3. PESSOAS ENVOLVIDAS.....</b>	<b>3</b>
<b>4. LOCAL E FERRAMENTAS DE TESTES.....</b>	<b>4</b>
4.1. Ferramentas de testes.....	4
4.2. Ambiente de testes.....	4
<b>5. RECURSOS NECESSÁRIOS.....</b>	<b>4</b>
<b>6. CRITÉRIOS USADOS.....</b>	<b>5</b>
<b>7. METODOLOGIA DE TESTE.....</b>	<b>6</b>
7.1. Etapas de Teste.....	6
7.2. Triagem de Bugs.....	8
7.3. Conclusão do Teste.....	9
<b>8. RESULTADOS DE TESTE.....</b>	<b>9</b>
<b>9. CRONOGRAMA.....</b>	<b>10</b>

---

## 1. INTRODUÇÃO

Este documento tem como objetivo apresentar o planejamento e a descrição das atividades relacionadas à execução dos testes no sistema Registra, uma plataforma web voltada para o controle e acompanhamento de estágios supervisionados obrigatórios no ensino superior. O plano de testes visa assegurar que todas as funcionalidades do sistema atendam aos requisitos estabelecidos, buscando garantir a confiabilidade, usabilidade e desempenho da aplicação.

## 2. RESUMO

Este plano de teste tem como objetivo validar o sistema Registra, uma plataforma web para controle de frequência de estágios supervisionados. Os testes serão conduzidos com base nos requisitos funcionais e não funcionais descritos na documentação oficial do projeto. A abordagem adotada será incremental, permitindo que os testes acompanhem o avanço do desenvolvimento. Serão utilizados testes manuais e automatizados para garantir a qualidade, segurança e usabilidade do sistema. As funcionalidades principais a serem testadas incluem cadastros, registros de frequência e aprovações. A execução será dividida por tipo de teste e priorizada conforme criticidade. Os resultados obtidos orientarão ajustes e melhorias antes da entrega final.

## 3. PESSOAS ENVOLVIDAS

A equipe responsável pelos testes do sistema Registra será composta por integrantes da própria equipe de desenvolvimento, que, neste projeto, também assumirão o papel de analistas de qualidade (QA).

- IVERTON EMIQUISON RIBEIRO DE BESSA – Analista de Testes (QA) e Desenvolvedor Front-end
- MATHEUS MARQUES NUNES – Analista de Testes (QA) e Desenvolvedor Back-end

## 4. LOCAL E FERRAMENTAS DE TESTES

### 4.1. Ferramentas de testes

Ferramenta	Descrição da ferramenta
NUnit	Framework para criação e execução de testes unitários e de integração
Selenium	Ferramenta para automação de testes funcionais em aplicações web
SonarLint	Extensão para análise estática de código e detecção de problemas de estilo
DevTools	Ferramenta nativa dos navegadores para inspeção de layout e responsividade
Lighthouse	Ferramenta para auditoria de desempenho, acessibilidade e boas práticas

### 4.2. Ambiente de testes

Máquina	Processador	Armazenamento	Sistema Operacional
HP 250 G8 Notebook PC	Intel Core i7-1165G7 (11ª Geração) @ 2.80GHz	238 GB SSD e 16 GB	Windows 11

## 5. RECURSOS NECESSÁRIOS

A realização dos testes do sistema Registra exigirá alguns recursos técnicos e humanos, organizados da seguinte forma:

- 
- **Pessoal:** Dois membros da equipe atuarão na função de testadores, com conhecimento básico em práticas de teste e entendimento funcional do sistema.
  - **Ambiente de Testes:** Cada testador utilizará sua própria máquina, previamente configurada com as dependências necessárias para execução do sistema e das ferramentas de teste.
  - **Ferramentas de Teste:** Serão utilizadas ferramentas como Jest (para testes unitários), Cypress ou Playwright (para testes funcionais), SonarLint (análise estática), DevTools (responsividade) e Lighthouse (auditoria de desempenho e acessibilidade).
  - **Acesso ao sistema Registra:** Será necessário um ambiente estável (versão de homologação) para execução dos testes completos.
  - **Documentação técnica:** Os testadores deverão ter acesso à documentação de requisitos, regras de negócio e fluxos do sistema.

## 6. CRITÉRIOS USADOS

Para garantir uma avaliação completa e eficaz do sistema Registra, os seguintes critérios foram definidos para guiar a execução dos testes:

**Cobertura mínima de testes estruturais:** Almeja-se atingir pelo menos 75% de cobertura de código nos testes unitários e de integração.

**Quantidade de casos de teste funcionais:** Devem ser elaborados e executados no mínimo 30 casos de teste, contemplando fluxos válidos e inválidos para funcionalidades essenciais (ex: cadastro, login, frequência).

**Critérios de aceitação por tipo de teste:**

- **Testes unitários:** validação de funções isoladas com assertivas diretas.
- **Testes de integração:** verificação da comunicação entre módulos como cadastro de estágio e aprovação de frequência.
- **Testes funcionais:** simulação de fluxos reais, cobrindo diferentes perfis (aluno, coordenador, supervisor).

---

**Análise de qualidade de código:** Espera-se identificar ao menos 15 alertas relevantes via SonarLint, priorizando aqueles de nível crítico ou alto.

**Testes de responsividade:** Devem ser identificados no mínimo 10 problemas de adaptação visual, priorizando os que afetam diretamente a usabilidade.

**Testes de portabilidade:** A aplicação deverá funcionar corretamente nos navegadores Chrome, Firefox e Edge, com comportamento consistente entre eles.

**Mensagens de erro e retorno do sistema:** Toda ação executada deve apresentar feedback claro ao usuário, tanto em casos de sucesso quanto de falha.

## 7. METODOLOGIA DE TESTE

A abordagem adotada será incremental, permitindo que os testes acompanhem a evolução das funcionalidades ao longo do desenvolvimento do sistema. Os testes serão aplicados em diferentes níveis, com foco em identificar falhas, validar regras de negócio e garantir que o sistema atenda aos requisitos definidos.

### 7.1. Etapas de Teste

Os testes serão organizados nos seguintes níveis:

- **Teste de Unidade**
  - Objetivo: Verificar se funções e métodos isolados estão se comportando conforme o esperado.
  - Ferramenta: Jest (para backend em JavaScript/TypeScript) ou ferramentas equivalentes.
  - Execução: Automatizada.
  - Critério de aceitação: 80% de cobertura mínima nos arquivos testados.
- **Teste de Integração**
  - Objetivo: Verificar se os módulos do sistema interagem corretamente entre si (ex: cadastro de estágio + aprovação + registro de frequência).
  - Execução: Automatizada.
  - Critério de aceitação: 70% de cobertura nos fluxos mais críticos.
- **Teste de Sistema**

- 
- Objetivo: Avaliar o funcionamento do sistema como um todo, simulando a experiência de cada perfil de usuário.
  - Execução: Manual.
  - Critério de aceitação: 75% dos testes aprovados com base nos requisitos definidos.
- **Teste de Portabilidade**
    - Objetivo: Verificar o funcionamento da aplicação nos navegadores Chrome, Firefox e Edge.
    - Ferramenta: Cypress ou Playwright.
    - Execução: Automatizada.
    - Critério de aceitação: Comportamento consistente em todos os navegadores testados.
- **Teste de Responsividade**
    - Objetivo: Garantir que a interface do sistema se adapte corretamente a diferentes tamanhos de tela (desktop, tablet, mobile).
    - Ferramenta: DevTools (Chrome).
    - Execução: Manual.
    - Critério de aceitação: No máximo 10 problemas visuais com impacto mínimo em usabilidade.
- **Teste de Qualidade de Página**
    - Objetivo: Avaliar desempenho, acessibilidade e boas práticas da aplicação.
    - Ferramenta: Lighthouse.
    - Execução: Automatizada.
    - Critério de aceitação: Pontuação mínima de 80% em pelo menos 3 dos 4 pilares da ferramenta.
- **Teste de Qualidade de Código**

- 
- Objetivo: Detectar problemas de legibilidade, padronização e boas práticas no código.
  - Ferramenta: SonarLint.
  - Execução: Automatizada.
  - Critério de aceitação: Identificar e corrigir ao menos 15 problemas relevantes.

## 7.2. Triagem de Bugs

Durante a execução dos testes, todos os erros identificados serão analisados, classificados e registrados para garantir sua rastreabilidade e correção adequada. A triagem seguirá as seguintes etapas:

### 1. Identificação

Os bugs serão detectados manualmente ou por ferramentas automatizadas durante os testes. Sempre que possível, o erro deverá ser reproduzido, documentado e evidenciado (print, log ou vídeo curto).

### 2. Classificação por Prioridade

Cada bug será classificado de acordo com seu impacto no sistema:

- Crítica: Impede o funcionamento de funcionalidades essenciais ou compromete a integridade dos dados. Exige correção imediata.  
Exemplo: falha no login, cadastro de frequência não sendo salvo.
- Alta: Afeta funcionalidades importantes, mas não bloqueia completamente o uso do sistema.  
Exemplo: erro na validação de formulário que impede o envio.
- Média: Problemas que não afetam diretamente a execução do sistema, mas comprometem a experiência do usuário.  
Exemplo: mensagens de erro mal posicionadas ou digitação errada na interface.
- Baixa: Erros cosméticos ou de interface sem impacto funcional.  
Exemplo: alinhamento de ícones, espaçamento visual.

### 3. Registro

Todos os bugs identificados serão documentados em planilha ou ferramenta de controle.

### 4. Acompanhamento e Correção



---

Os bugs serão revisados conforme prioridade e tratados entre as entregas de sprints ou ciclos de desenvolvimento. A equipe fará a revalidação após cada correção para garantir a resolução do problema.

### 7.3. Conclusão do Teste

Ao final da execução dos testes, será feita uma avaliação geral para verificar se o sistema Registra atende aos requisitos definidos. A conclusão será baseada nos seguintes critérios:

- **Cobertura mínima de testes:** Espera-se atingir pelo menos 75% de cobertura nos testes planejados, incluindo casos unitários, de integração e funcionais.
- **Correção de bugs críticos:** Todos os erros classificados como críticos ou alta prioridade deverão estar corrigidos e revalidados antes da entrega final.
- **Registro dos resultados:** Os resultados dos testes (casos aprovados e reprovados) serão documentados, permitindo rastreabilidade e análise de melhoria contínua.
- **Sugestões de melhoria:** Com base nas falhas encontradas, serão sugeridas melhorias no sistema, tanto em aspectos técnicos quanto de usabilidade.

A conclusão dos testes marcará o início da fase de manutenção preventiva, onde eventuais ajustes serão aplicados antes da versão final ser considerada pronta para uso.

## 8. RESULTADOS DE TESTE

Após a execução dos testes planejados, será elaborado um relatório consolidando os resultados obtidos, com o objetivo de verificar o grau de conformidade do sistema em relação aos requisitos definidos.

Os principais pontos que serão documentados incluem:

- **Casos de teste executados:** Quantidade total de casos aplicados, divididos entre testes unitários, de integração e funcionais.
- **Percentual de aprovação:** Número de testes aprovados em relação ao total executado.
- **Falhas encontradas:** Descrição dos principais bugs detectados, com destaque para os classificados como críticos ou de alta prioridade.
- **Requisitos atendidos x pendentes:** Avaliação sobre quais funcionalidades estão completas, parcialmente implementadas ou não funcionam como esperado.

- **Desempenho e responsividade:** Análise baseada nas ferramentas Lighthouse e DevTools, indicando possíveis gargalos ou melhorias visuais.
- **Sugestões de melhoria:** Anotações sobre pontos que podem ser ajustados além da correção de erros, visando otimizar a experiência do usuário e a eficiência do sistema.

## 9. CRONOGRAMA

Tarefa	Responsável(es)	Status	Início	Término
Planejamento dos testes	Iverton, Matheus	Concluída	14/05/2025	16/05/2025
Execução dos testes unitários (Jest)				
Execução dos testes unitários (Jest)				
Execução dos testes funcionais (Cypress)				
Testes de responsividade com DevTools				
Testes de qualidade com Lighthouse e SonarLint				
Documentação dos resultados finais				