UNIVERSIDADE DE SÃO PAULO — USP
BACHARELADO EM SISTEMAS DE INFORMAÇÃO
PROJETO E DESENVOLVIMENTO DE SISTEMAS DE INFORMAÇÃO

SSC0536 - Projeto e Desenvolvimento de Sistemas de Informação

Sistema Web para Registro Digital de Ocorrências da Defesa Civil de São Carlos (SiG-DC São Carlos).

Arthur Moreira Correa	13749952
Eduardo Ribeiro Rodrigues	13696679
Murilo Genghi Rossi	14599915
Vinicius Moraes de Paiva	13783530
Vitor Zago	14783595

São Carlos–SP
2025

Plano de Testes de Integração e Sistema - SiG-DC São Carlos

1. Introdução

Este documento detalha a estratégia e os cenários para os Testes de Integração e Testes de Sistema do projeto SiG-DC São Carlos. O objetivo principal desta fase é validar a interação entre os diferentes componentes do software e garantir que o sistema, na totalidade, atende aos requisitos de negócio e às expectativas do cliente, operando de forma coesa, estável e confiável.

O processo de testes seguirá uma abordagem metódica para identificar e corrigir defeitos antes da fase de aceitação final pelo cliente e da implantação em produção.

2. Estratégia Geral de Testes

A estratégia será dividida em duas grandes fases sequenciais:

1. **Testes Unitários (Base da Pirâmide):** Focados em verificar as menores partes isoladas do código (funções, métodos) para garantir que funcionem como esperado. Serão a principal ferramenta para garantir a qualidade interna do código.
2. **Testes de Integração:** Focados em verificar a comunicação e a troca de dados entre os principais componentes da arquitetura: o Front-end (interface web), o Back-end (API NestJS) e o Banco de Dados (Firebase Firestore). Esta fase é de responsabilidade primária da equipe de desenvolvimento.
3. **Testes de Sistema:** Focados em validar o sistema completo sob a perspectiva do usuário final. Serão executados fluxos de trabalho completos ("end-to-end") para garantir que todas as funcionalidades se comportam conforme o esperado em um ambiente que simula a produção. Esta fase será conduzida pela equipe de testes do projeto.

3. Testes Unitários

O objetivo é garantir a correção lógica de cada componente individual do sistema, principalmente na camada de Back-end. Cada função ou método que contém lógica de negócio (validações, cálculos, transformações de dados) deve ser testado de forma isolada para verificar seu comportamento em diferentes cenários.

Os testes unitários serão automatizados e escritos pelos próprios desenvolvedores à medida que implementam novas funcionalidades. Utilizaremos o framework Jest, que é o padrão para o ecossistema NestJS, permitindo a criação de suítes de teste claras e eficientes. Os testes serão focados nos services e controllers da aplicação Back-end para validar as regras de negócio.

Como serão mostrados e verificados: A verificação da qualidade será contínua e automatizada:

- **Integração Contínua (CI):** Os testes unitários serão executados automaticamente a cada *push* de código para o repositório no GitHub, através do **GitHub Actions**. Nenhum código que quebre os testes existentes poderá ser integrado à branch principal.

- **Relatório de Cobertura de Código (Code Coverage):** A execução dos testes gerará um relatório de cobertura, indicando a porcentagem do código-fonte que está sendo validada pelos testes. Este relatório será analisado para garantir que novas funcionalidades tenham uma cobertura mínima de testes, servindo como um indicador de qualidade. O resultado será visível diretamente nos logs da execução do GitHub Actions.

4. Testes de Integração

O objetivo é garantir que o Front-end consiga consumir a API do Back-end de forma correta e que o Back-end, por sua vez, persista e recupere os dados do Firebase Firestore de maneira íntegra.

Os testes serão executados em um ambiente de desenvolvimento local, utilizando o Firebase Emulators para simular o banco de dados e os serviços de autenticação. A equipe irá interagir com a interface web e, simultaneamente, monitorar as requisições de API e as operações no banco de dados para validar o fluxo de dados.

Cenários de Teste de Integração

ID	**Cenário de Integração**	**Passos de Execução**	**Resultado Esperado**
INT-01	Autenticação de Usuário	1. Front-end envia credenciais (email/senha) para a API de login. 2. Back-end valida as credenciais com o Firebase Authentication.	O Back-end retorna um token de acesso válido, e o Front-end redireciona o usuário para o Dashboard.
INT-02	Criação de Nova Ocorrência	1. Front-end envia os dados do formulário para o endpoint de criação da API. 2. Back-end valida os dados e os estrutura. 3. Back-end insere um novo documento na coleção ocorrencias no Firestore.	A API retorna um status de sucesso (201 Created). Um novo documento de ocorrência é criado no Firestore com os dados corretos.

INT-03	Listagem de Ocorrências	1. Front-end solicita a lista de ocorrências ao endpoint da API. 2. Back-end consulta a coleção ocorrencias no Firestore. 3. Back-end retorna a lista de ocorrências para o Front-end.	O Front-end recebe um array de objetos de ocorrência e exibe a lista corretamente no Dashboard.

5. Testes de Sistema

O objetivo é validar o sistema completo do ponto de vista funcional e não funcional, simulando o uso real por um agente da Defesa Civil. O foco é na experiência do usuário e na conformidade com os requisitos de negócio.

Os testes serão conduzidos manualmente pela equipe designada, seguindo os cenários descritos abaixo. A equipe acessará a aplicação web a partir de diferentes dispositivos (desktop e mobile) para verificar a responsividade e a usabilidade em diversos contextos.

Cenários de Teste de Sistema

ID	**Cenário de Uso (User Story)**	**Pré-condições**	**Passos de Execução**	**Resultado Esperado**
SIS-01	Acesso ao Sistema	- Ter um usuário "agente" cadastrado e ativo.	1. Acessar a URL do sistema. 2. Inserir um email inválido e senha válida. 3. Inserir um email válido e senha inválida. 4. Inserir email e senha válidos.	2. O sistema exibe a mensagem "Usuário ou senha inválidos". 3. O sistema exibe a mensagem "Usuário ou senha inválidos". 4. O login é bem-sucedido e o Dashboard é exibido.

SIS-02	Registro de Ocorrência	- Usuário "agente" logado no sistema.	1. Clicar em "Registrar Nova Ocorrência". 2. Preencher todos os campos obrigatórios do formulário. 3. Clicar em "Salvar".	A ocorrência é salva com sucesso. O sistema redireciona para o Dashboard, onde a nova ocorrência aparece no topo da lista com o status "Aberta".
SIS-03	Busca e Filtragem	- Usuário logado. - Existirem múltiplas ocorrências cadastradas com status e tipos diferentes.	1. No Dashboard, usar o campo de busca para procurar por um protocolo específico. 2. Limpar a busca. 3. Usar o filtro de "Status" e selecionar "Em atendimento".	1. Apenas a ocorrência com o protocolo pesquisado é exibida. 3. A lista é atualizada e exibe apenas as ocorrências com o status "Em atendimento".
SIS-04	Atualização de Status	- Usuário "coordenador" logado. - Existir uma ocorrência com status "Aberta".	1. No Dashboard, clicar em uma ocorrência "Aberta". 2. Na tela de detalhes, alterar o status para "Finalizada". 3. Adicionar uma nota no histórico de atualizações. 4. Salvar as alterações.	O status da ocorrência é atualizado para "Finalizada". A nota é adicionada ao histórico. A informação é refletida corretamente no Dashboard.
SIS-05	Teste de Responsividade	- Usuário logado.	1. Acessar o sistema em um navegador de desktop e redimensionar a janela.	1. A interface se adapta ao tamanho da tela sem quebras de layout. 2. O layout está

			2. Acessar o sistema através de um navegador em um smartphone. 3. Tentar registrar uma nova ocorrência pelo smartphone.	otimizado para telas pequenas, com boa legibilidade e botões fáceis de tocar. 3. O formulário é totalmente funcional e usável no dispositivo móvel.