UFERSA
Universidade Federal Rural do Semi-Árido
Centro Multidisciplinar de Pau dos Ferros – CMPF
Bacharelado em Tecnologia da Informação

Plano de Teste: FinanSee

IDENTIFICAÇÃO DO PROJETO

Projeto

- FinanSee

Docente

- Alysson Filgueira Milanez

Equipe

- Antonio Welles Queiroz de Paiva
- Carlos Danniel Gonçalves da Silva
- Cicero Araújo Rodrigues
- Fernando Umbilino Alves
- Lidiana Costa de Souza
- Lucas Mairon Oliveira Camilo

HISTÓRICO DE ALTERAÇÕES

Versão	Data	Autor	Descrição
1.0	24/06/2025	Lidiana Costa Cicero Araujo Carlos Danniel	Plano de teste inicial

SUMÁRIO

1. INTRODUÇÃO

Este documento tem como objetivo apresentar o planejamento e a descrição das atividades relacionadas à execução dos testes no sistema FinanSee, uma plataforma web voltada para o acompanhamento e organização de despesas visando melhorar a organização financeira dos usuários. 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 FinanSee, uma plataforma web para organização financeira. Os testes serão conduzidos com base nos requisitos funcionais e não funcionais descritos na especificação de requisitos do projeto. A abordagem adotada é o modelo 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 cadastro e visualização de despesas, visualização de gráficos e resumos, como também o gerenciamento dos usuários. As funcionalidades a serem testadas são as que forem desenvolvidas. Os resultados obtidos orientarão ajustes e melhorias antes da entrega final.

3. PESSOAS ENVOLVIDAS

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

- Lidiana Costa de Souza – Analista de Testes (QA) e Desenvolvedor Back-end

4. LOCAL E FERRAMENTAS DE TESTES

4.1. Ferramentas de testes

Ferramenta	Descrição da ferramenta
Pytest/Unittest	Framework para criação e execução de testes unitários e de integração
Selenium	Ferramenta para automação de testes funcionais em aplicações web
Pylint	Extensão para análise estática de código.
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
Aspire A515-45 V1.52 Notebook	AMD Ryzen 7 5700U with Radeon G	512GB SSD e 8GB	Ubuntu 24.04.2 LTS x86_64

5. RECURSOS NECESSÁRIOS

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

- **Pessoal:** Um membro da equipe atuará na função de testador, 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 Pytest/Unittest(para testes unitários), Pytest/Unittest ou Selenium (para testes funcionais), Pylint(análise estática), DevTools (responsividade) e Lighthouse (auditoria de desempenho e acessibilidade).

- **Acesso ao sistema FinanSee:** Será necessário um ambiente estável (versão de homologação) para execução dos testes completos.

- **Documentação técnica:** Os testadores deverão ter acesso à documentação de especificação de requisitos, documento de detalhamento do projeto e documentação de especificação formal.

6. CRITÉRIOS USADOS

Para garantir uma avaliação completa e eficaz do sistema FinanSee, 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, cadastro de despesas).

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 despesas e gerenciamento de categoria.
- ● **Testes funcionais:** simulação de fluxos reais, cobrindo diferentes as funcionalidades concluídas

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: Pytest/Unittet.
    - 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 despesa + vincular a período + vincular categoria).
    - 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: Pylint.
  - ○ 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	Lidiana Costa	Concluída	24/06/2025	24/06/2025
Execução dos testes unitários (Pytest)				
Execução dos testes unitários (Pytest)				
Testes de responsividade com DevTools				
Testes de qualidade com Lighthouse				
Documentação dos resultados finais				