

Relatório de Testes

Lit Quest [1.0]

Profº Tiago Funk

Clara dos Santos Becker

Lucas Gabriel Falcade Nunes

1. INTRODUÇÃO

Este documento é destinado à elaboração de um plano de teste para o sistema LitQuest, uma plataforma de vendas de livros online que combina e-commerce com personalização de metas de leitura. O sistema foi projetado para engajar leitores, com a adição de metas de leitura mensais e semanais. Bem como, facilitar a escolha da próxima compra realizada pelo usuário e proporcionar simplicidade na utilização das ferramentas de compra do site.

Se faz necessário o uso de um sistema que seja intuitivo e personalizável, além de focado nas metas para manter o usuário sempre interessado na sua próxima leitura. Também, é necessário abranger o que foi registrado no histórico de compras, histórico e filtro de buscas de obras literárias. O link de acesso a todos os códigos de testes estão em <https://github.com/lucasgfn/PIN2>.

2. REQUISITOS A SEREM TESTADOS

Tabela 1. Requisitos Funcionais

ID	Descrição	Impacto	Prioridade
01	O Sistema deve permitir ao usuário manter perfil como comprador/usuário.	8	Alta
02	O Sistema deve realizar a autenticação dos usuários.	8	Alta
03	O Sistema deve ter uma página de detalhes sobre o livro que foi consultado.	6	Média
04	O Sistema deve possibilitar ao usuário personalização de metas de leitura mensal.	6	Média
05	O Sistema deve possibilitar ao usuário personalização das metas de páginas lidas diariamente.	6	Média
06	O Sistema deve possibilitar ao usuário buscar livros os livros cadastrador	6	Média
07	O Sistema deve exibir corretamente os dados completos do livro cadastrado	6	Alta
08	O Sistema deve possibilitar manter os livros e autores, assim como o vínculo correto entre as entidades.	8	Alta
09	O Sistema deve permitir manter o carrinho de compras com os itens (livros)	8	Alta
10	O Sistema deve realizar o cálculo corretamente dos itens levando em	8	Alta

	consideração a quantidade e o valor unitário.		
--	---	--	--

Tabela 2. Requisitos de Usabilidade

ID	Descrição	Eficácia	Eficiência	Satisfação
01	O Sistema deve ser de fácil utilização e intuitivo para os usuários.	90% consegue entender o sistema	Aprende em no máximo 1 hora	90% de satisfação

3. PROJETO DE TESTES

Nível de Teste	Unitário
Técnicas de Teste	Caixa Cinza
Critérios de seleção de entradas	Baseados em Casos de Uso Baseado em Fluxo de Controle Análise Valor Limite
Ferramentas	JUnit, Jakarta Bean Validation, Assert J

Nível de Teste	Integração
Técnicas de Teste	Caixa Preta
Critérios de seleção de entradas	Baseados em Casos de Uso
Ferramentas	JUnit, Spring Test/RestTemplate, AssertJ, Spring Boot Test

Nível de Teste	Sistema
Técnicas de Teste	Caixa Preta
Critérios de seleção de entradas	Baseados em Casos de Uso
Ferramentas	Selenium

Nível de Teste	Aceitação
Técnicas de Teste	SUS - <i>System Usability Scale</i>
Critérios de seleção de entradas	Valores corretos, buscando averiguar Efetividade, Eficiência e Satisfação.
Ferramentas	Formulário

4. CENÁRIOS DE TESTE

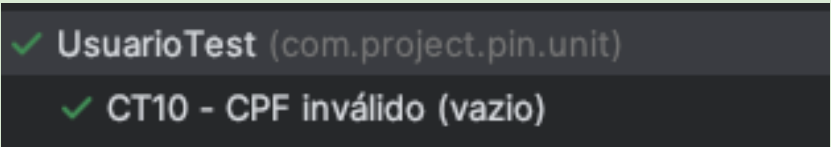
4.1. TESTES UNITÁRIOS

RF01	O Sistema deve permitir ao usuário manter perfil como comprador/usuario.
RT01	Validar os campos da entidade <i>Usuario</i>
Objetivo	Garantir que os campos da entidade <i>Usuario</i> sejam validados corretamente conforme as restrições definidas, incluindo limites de tamanhos e formatos.
Localização	Realiza o teste da Entidade Usuário
Pré-condições	1. Entidade <i>Usuario</i> com as anotações de validação corretas @Size, @NotBlank, @Email, @Pattern.
Passo a passo	1. Criar as instâncias com o objeto construtor UsuarioTeste de cada Caso de Teste (CT) utilizando Análise de Valores Limites. 2. Validar objeto usando <code>validator.validate()</code>
Nível de teste	Unitário
Técnica de teste	Caixa Cinza
Critério de seleção de entradas	<i>Baseado nas restrições de validade da entidade.</i>
CT01	Valida senha com 10 caracteres
Entrada	<ul style="list-style-type: none">• Nome: Teste• Email: <u>teste@email.com</u>• CPF: 123.456.789-00• Username: usuario1• Password: 1234567890
Resultado esperado	Senha válida e sem gerar violações.
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Usuario</i> são validados corretamente antes de qualquer persistência.





Resultado Obtido	<div><div>✓ UsuarioTest (com.project.pin.unit)</div><div>✓ CT01 - Senha com tamanho mínimo válido (10)</div></div>
Classificação	Aprovado!
CT02	Valida senha 200 caracteres
Entrada	<ul style="list-style-type: none">Nome: TesteEmail: <u>teste@email.com</u>CPF: 123.456.789-00Username: usuario2Password: aa aa aa aa aaaaaaaaaaaa
Resultado esperado	Senha válida e sem gerar violações.
Prioridade	Alta
Pós-condições	Os objetos da entidade <i>Usuario</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div><div>✓ UsuarioTest (com.project.pin.unit)</div><div>✓ CT02 - Senha com tamanho máximo válido (200)</div></div>
Classificação	Aprovado!
CT03	Valida senha com tamanho abaixo do mínimo permitido
Entrada	<ul style="list-style-type: none">Nome: TesteEmail: <u>teste@email.com</u>CPF: 123.456.789-02Username: usuario3Password: 123456789
Resultado esperado	Deve gerar uma <i>Constraint Violation</i> no campo <i>password</i> .
Prioridade	Alta
Pós-condições	Os objetos da entidade <i>Usuario</i> são validados corretamente antes de

Pós-condições	Os objetos da entidade <i>Usuario</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> <div>✓ UsuarioTest (com.project.pin.unit)</div> <div>✓ CT05 - Username válido (sem espaços)</div> </div>
Classificação	<i>Aprovado!</i>
CT06	Verifica se o username é válido (com espaços/inválido)
Entrada	<ul style="list-style-type: none"> Nome: Teste Email: <u>teste@email.com</u> CPF: 123.456.789-11 Username: user teste Password:1234567890
Resultado esperado	<i>Deve gerar uma Constraint Violation no campo username.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Usuario</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> <div>✓ UsuarioTest (com.project.pin.unit)</div> <div>✓ CT06 - Username inválido (contendo espaços)</div> </div>
Classificação	<i>Aprovado!</i>
CT07	Verifica se o e-mail está no padrão válido (inválido)
Entrada	<ul style="list-style-type: none"> Nome: Teste Email: invalido-email CPF: 123.456.789-12 Username: user teste Password:1234567890
Resultado esperado	<i>Deve gerar uma Constraint Violation no campo email.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Usuario</i> são validados corretamente antes de qualquer persistência.

Resultado Obtido	<div> <div>✓ UsuarioTest (com.project.pin.unit)</div> <div>✓ CT07 - Email inválido</div> </div>
Classificação	Aprovado!
CT08	Verifica se o e-mail está no padrão válido (válido)
Entrada	<ul style="list-style-type: none"> Nome: Teste Email: valido@email.com CPF: 123.456.789-11 Username: user1 Password:1234567890
Resultado esperado	E-mail válido e sem gerar violações.
Prioridade	Alta
Pós-condições	Os objetos da entidade <i>Usuario</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> <div>✓ UsuarioTest (com.project.pin.unit)</div> <div>✓ CT08 - Email válido</div> </div>
Classificação	Aprovado!
CT09	Verifica se o cpf está no padrão válido (não vazio)
Entrada	<ul style="list-style-type: none"> Nome: Teste Email: teste@email.com CPF: 123.456.789-30 Username: user1 Password:1234567890
Resultado esperado	CPF válido e sem gerar violações.
Prioridade	Alta
Pós-condições	Os objetos da entidade <i>Usuario</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> <div>✓ UsuarioTest (com.project.pin.unit)</div> <div>✓ CT09 - CPF válido (não vazio)</div> </div>

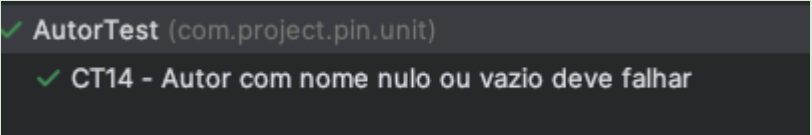
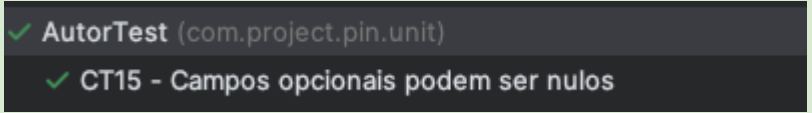
Classificação	<i>Aprovado!</i>
CT10	Verifica se o cpf está no padrão válido (vazio/inválido)
Entrada	<ul style="list-style-type: none"> • Nome: Teste • Email: teste@email.com • CPF: • Username: user1 • Password: 1234567890
Resultado esperado	<i>Deve gerar uma Constraint Violation no campo cpf.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Usuario</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	
Classificação	<i>Aprovado!</i>

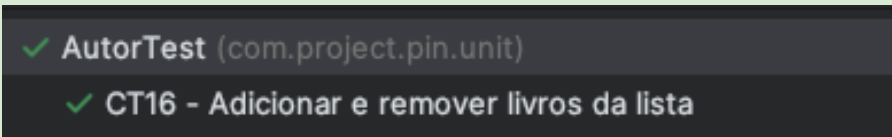
RF01	O Sistema deve permitir ao usuário manter perfil como comprador.
RT02	Validar os campos da entidade <i>Comprador</i>
Objetivo	Garantir que os campos da entidade <i>Comprador</i> sejam validados corretamente conforme as restrições definidas, incluindo limites de tamanhos e formatos.
Localização	Realiza o teste da Entidade Comprador
Pré-condições	1. Entidade Comprador com as anotações de validação corretas <code>@NotBlank</code> e <code>@NotNull</code>
Passo a passo	1. Criar as instâncias com o objeto Comprador para cada Caso de Teste (CT) utilizando Análise de Valores Limites. 2. Validar objeto usando <code>validator.validate()</code>
Nível de teste	Unitário
Técnica de teste	Caixa Cinza
Critério de	<i>Baseado nas restrições de validade da entidade.</i>

seleção de entradas	
CT11	Validar Telefone Válido (não vazio)
Entrada	<ul style="list-style-type: none"> • Nome: Lucas • Email: lucas@email.com • CPF: 123.456.789-00 • Username: lucas • Password: 1234567890 • Telefone: 123456789
Resultado esperado	Telefone válido e sem gerar violações.
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Comprador</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div>  CompradorTest (com.project.pin.unit)  CT11 - Telefone válido (não vazio) </div>
Classificação	<i>Aprovado!</i>
CT12	Validar Telefone Válido (inválido)
Entrada	<ul style="list-style-type: none"> • Nome: Lucas • Email: lucas@email.com • CPF: 123.456.789-01 • Username: lucas • Password: 1234567890 • Telefone:
Resultado esperado	<i>Deve gerar uma Constraint Violation no campo Telefone.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Comprador</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div>  CompradorTest (com.project.pin.unit)  CT12 - Telefone inválido (vazio) </div>

Classificação	<i>Aprovado!</i>
----------------------	------------------

RF08	O Sistema deve possibilitar manter os livros e autores, assim como o vínculo correto entre as entidades.
RT03	Validar os campos da entidade <i>Autor</i>
Objetivo	Garantir que os campos da entidade <i>Autor</i> sejam validados corretamente conforme as restrições definidas, incluindo limites de tamanhos e formatos.
Localização	Realiza o teste da Entidade <i>Autor</i>
Pré-condições	1. Entidade Comprador com as anotações de validação corretas <code>@NotBlank</code> .
Passo a passo	1. Criar as instâncias com o objeto <i>Autor</i> para cada Caso de Teste (CT) utilizando Análise de Valores Limites. 2. Validar objeto usando <code>validator.validate()</code>
Nível de teste	Unitário
Técnica de teste	Caixa Cinza
Critério de seleção de entradas	<i>Baseado nas restrições de validade da entidade.</i>
CT13	Validar Objeto Autor Válido com todos os atributos preenchidos
Entrada	<ul style="list-style-type: none"> Nome: <code>Raphael Montes</code> Sobre: <code>Romancista policial brasileiro</code> Imagem: <code>https://img.com/foto.jpg</code>
Resultado esperado	Objeto Autor válido e sem gerar violações.
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Autor</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> ✓ AutorTest (com.project.pin.unit) </div> <div> ✓ CT13 - Autor válido com todos os campos preenchidos </div>
Classificação	<i>Aprovado!</i>

CT14	Validar Autor com o nome nulo ou vazio
Entrada	<ul style="list-style-type: none"> • Nome : • Sobre: Romancista policial brasileiro • Imagem: https://img.com/foto.jpg
Resultado esperado	<i>Deve gerar uma Constraint Violation no campo Nome.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Autor</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	
Classificação	<i>Aprovado!</i>
CT15	Validar Autor com os campos opcionais nulos
Entrada	<ul style="list-style-type: none"> • Nome: Autor Teste • Sobre: • Imagem:
Resultado esperado	Objeto Autor válido e sem gerar violações.
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Autor</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	
Classificação	<i>Aprovado!</i>
CT16	Adicionar e remover Livros da Lista
Entrada	<ul style="list-style-type: none"> • Autor: Autor Teste • Nome do livro do autor: Livro 1 • Nome do livro do autor: Livro 2


Resultado esperado	A lista de livros do objeto Autor é válida e não gerar violações. Podemos adicionar e remover os livros da lista.
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Autor</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	
Classificação	<i>Aprovado!</i>

RF07	O Sistema deve exibir corretamente os dados completos do livro cadastrado
RT04	Validar os campos da entidade <i>Livro</i>
Objetivo	Garantir que os campos da entidade <i>Livro</i> sejam validados corretamente conforme as restrições definidas, incluindo limites de tamanhos e formatos.
Localização	Realiza o teste da Entidade <i>Livro</i>
Pré-condições	1. Entidade <i>Livro</i> com as anotações de validação corretas @NotBlank , @Min , @Max , @Positive .
Passo a passo	<ol style="list-style-type: none"> 1. Criar as instâncias do Autor para vínculo ao livro (obrigatório) 2. Criar um método privado <code>criarLivroValido()</code> para criar o livro com todos os dados solicitados. 3. Realizar a mudança nos métodos de teste para utilizando o <code>Set()</code>. 4. Validar objeto usando <code>validator.validate()</code>.
Nível de teste	Unitário
Técnica de teste	Caixa Cinza
Critério de seleção de entradas	<i>Baseado nas restrições de validade da entidade.</i>
CT17	Validar o ISBN obrigatório - Inválido
Entrada	<ul style="list-style-type: none"> • ISBN: • Nome do livro: Livro Teste

	<ul style="list-style-type: none"> • Preço unitário: 50.0 • Páginas: 100 • Ano publicado: 2023 • Imagem: http://imagem.com/livro.jpg <ul style="list-style-type: none"> • Nome: Autor Teste • Sobre: Descrição do autor • Imagem: http://imagem.com/autor.jpg
Resultado esperado	<i>Deve gerar uma Constraint Violation no campo ISBN.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Livro</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> <div>✓ LivroTest (com.project.pin.unit)</div> <div>✓ CT17 - Valida ISBN obrigatório</div> </div>
Classificação	<i>Aprovado!</i>
CT18	Validar o Nome obrigatório - Inválido
Entrada	<ul style="list-style-type: none"> • ISBN: ISBN-1234 • Nome do livro: • Preço unitário: 50.0 • Páginas: 100 • Ano publicado: 2023 • Imagem: http://imagem.com/livro.jpg

	<ul style="list-style-type: none"> Nome: Autor Teste Sobre: Descrição do autor Imagem: http://imagem.com/autor.jpg
Resultado esperado	<i>Deve gerar uma Constraint Violation no campo Nome.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Livro</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> <div>✓ LivroTest (com.project.pin.unit)</div> <div>✓ CT18 - Valida nome do livro obrigatório</div> </div>
Classificação	<i>Aprovado!</i>
CT19	Validar o Preço com Valor Negativo - Inválido
Entrada	<ul style="list-style-type: none"> ISBN: ISBN-1234 Nome do livro: Livro Teste Preço unitário: Páginas: 100 Ano publicado: 2023 Imagem: http://imagem.com/livro.jpg Nome: Autor Teste Sobre: Descrição do autor Imagem: http://imagem.com/autor.jpg

Resultado esperado	<i>Deve gerar uma Constraint Violation no campo PrecoUnit.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Livro</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> <div>✓ LivroTest (com.project.pin.unit)</div> <div>✓ CT19 - Valida preço unitário positivo</div> </div>
Classificação	<i>Aprovado!</i>
CT20	Validar Valor Mínimo de Páginas - Inválido
Entrada	<ul style="list-style-type: none"> • ISBN: ISBN-1234 • Nome do livro: Livro Teste • Preço unitário: 50,0 • Páginas: • Ano publicado: 2023 • Imagem: http://imagem.com/livro.jpg • Nome: Autor Teste • Sobre: Descrição do autor • Imagem: http://imagem.com/autor.jpg
Resultado esperado	<i>Deve gerar uma Constraint Violation no campo Pages.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Livro</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> <div>✓ LivroTest (com.project.pin.unit)</div> <div>✓ CT20 - Valida número mínimo de páginas</div> </div>

Classificação	<i>Aprovado!</i>
CT21	Validar Ano Mínimo de Publicação - Inválido
Entrada	<ul style="list-style-type: none"> ISBN: ISBN-1234 Nome do livro: Livro Teste Preço unitário: 50,0 Páginas: 999 Ano publicado: 2023 Imagem: http://imagem.com/livro.jpg Nome: Autor Teste Sobre: Descrição do autor Imagem: http://imagem.com/autor.jpg
Resultado esperado	<i>Deve gerar uma Constraint Violation no campo AnoPublicado.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Livro</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	
Classificação	<i>Aprovado!</i>
CT22	Validar Ano Máximo de Publicação - Inválido
Entrada	<ul style="list-style-type: none"> ISBN: ISBN-1234 Nome do livro: Livro Teste Preço unitário: 50,0 Páginas: 100000

	<ul style="list-style-type: none"> Ano publicado: 2023 Imagem: http://imagem.com/livro.jpg Nome: Autor Teste Sobre: Descrição do autor Imagem: http://imagem.com/autor.jpg
Resultado esperado	<i>Deve gerar uma Constraint Violation no campo AnoPublicado.</i>
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Livro</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> ✓ LivroTest (com.project.pin.unit) ✓ CT22 - Valida ano publicado máximo </div>
Classificação	<i>Aprovado!</i>
CT23	Validar livro totalmente com dados corretos - Válido
Entrada	<ul style="list-style-type: none"> ISBN: ISBN-1234 Nome do livro: Livro Teste Preço unitário: 50,0 Páginas: 100 Ano publicado: 2023 Imagem: http://imagem.com/livro.jpg Nome: Autor Teste Sobre: Descrição do autor Imagem: http://imagem.com/autor.jpg

Resultado esperado	Objeto Livro válido e sem gerar violações.
Prioridade	<i>Alta</i>
Pós-condições	Os objetos da entidade <i>Autor</i> são validados corretamente antes de qualquer persistência.
Resultado Obtido	<div> ✓ LivroTest (com.project.pin.unit) ✓ CT23 - Valida livro totalmente válido </div>
Classificação	<i>Aprovado!</i>

4.2 TESTES DE INTEGRAÇÃO

RF01	Cadastro e Autenticação de Comprador
RT01	Integração dos Funcionalidades do Cadastro de Comprador e Autenticação de Comprador
Objetivo do roteiro de teste	Verificar se a integração entre as camadas permite cadastrar e autenticar corretamente um comprador, garantindo que os dados sejam persistidos no banco de dados.
Localização	Realizar teste de integração do sistema com a requisição <code>/comprador</code>
Pré-condições	Banco de dados ativo
Passo a passo	<ol style="list-style-type: none"> 1. Realizar cadastro do comprador na URL: <code>"/comprador"</code> 2. Efetuar login com os mesmos dados cadastrais URL: <code>"/auth/login"</code>
Nível de teste	Integração
Técnica de teste	Caixa Preta
Critério de seleção de entradas	Baseados em casos de uso
CT01	Cadastrar comprador
Entradas	<ul style="list-style-type: none"> • Nome: Lucas Falcade • Email: lucas.falcade@email.com • CPF: 123.456.789-11

	<ul style="list-style-type: none"> • Username: lucasfalcade • Senha: senhaSegura123 • Rua: Rua das Flores, 120 • Telefone: (48) 99999-8888 • Bairro: Bela Vista • Cidade: Ibirama • CEP: 88000-000 • Estado: SC • Imagem: perfil_lucas.jpg • Nível: 1 • Recebeu desconto: false
Resultados esperados	<i>Cadastro: Mensagem de sucesso após o cadastro</i>
Prioridade	Alta
Resultado Obtido	<div> <div>✓ CompradorIntegrationTest (com.project.pin.integra)</div> <div>✓ CT01 - Deve cadastrar comprador com sucesso - Cadastro com Sucesso</div> </div>
Classificação	Aprovado!
Pós-condições	<i>Comprador salvo no banco de dados e autenticado com sucesso</i>
CT02	Autenticar comprador
Entradas	<pre>{ "username": "lucasfalcade", "password": "senhaSegura123" }</pre>
Resultados esperados	<i>Autenticação: Sucesso no Login</i>
Prioridade	Alta
Pós-condições	<i>Autenticado com sucesso</i>

Resultado Obtido	<div> <div>✓ CompradorIntegrationTest (com.project.pin.integra)</div> <div>✓ CT01 - Deve cadastrar comprador com sucesso - Cadastro com Sucesso</div> <div>✓ CT02 - Deve autenticar comprador com sucesso após cadastro - Login bem-sucedido!</div> </div>
Classificação	<i>Aprovado!</i>

RF03	Integração entre o livro consultado e seu detalhamento
RT02	Integração dos Funcionalidades do Livro
Objetivo do roteiro de teste	Verificar se a integração da busca contém corretamente todos os dados de detalhamento do livro consultado ou busca por título.
Localização	Teste de Requisição: <code>/livros/buscar?nomeLivro={titulo}</code>
Pré-condições	O livro deve estar cadastrado no banco de dados.
Passo a passo	<ol style="list-style-type: none"> 1. Realizar o login como usuário 2. Consultar o livro por título 3. Selecionar o livro retornado para visualizar o detalhamento
Nível de teste	Integração
Técnica de teste	Caixa Preta
Critério de seleção de entradas	Baseados em casos de uso
CT03	Consulta detalhada de livro existente
Entradas	<ul style="list-style-type: none"> • GET /livros/buscar?nomeLivro=Uma Família Feliz
Resultados esperados	<i>O sistema deve retornar o status 200 (OK) e exibir todas as informações detalhadas do livro consultado.</i>
Prioridade	Alta
Pós-condições	<i>Usuário visualiza a página de detalhamento com todos os dados do livro consultado e selecionado.</i>
Resultado Obtido	<div> <div>✓ LivroIntegrationTest (com.project.pin.integra)</div> <div>✓ CT03 - Consulta detalhada de livro existente</div> </div>

	<pre>assertThat(detalhado.getNomeLivro()).isEqualTo(expected: "Uma Família Feliz"); assertThat(detalhado.getIsbn()).isEqualTo(expected: "ISBN-123456789"); assertThat(detalhado.getPages()).isEqualTo(expected: 320); assertThat(detalhado.getAnoPublicado()).isEqualTo(expected: 2021); assertThat(detalhado.getSinopse()).contains("família"); assertThat(detalhado.getPrecoUnit()).isEqualTo(expected: 59.90); assertThat(detalhado.getImg()).isEqualTo(expected: "familiafeliz.jpg");</pre>
Classificação	<i>Aprovado!</i>

RF04	Personalização de metas de leitura mensal
RT03	Integração entre MonthGoalController e GoalRepository
Objetivo do roteiro de teste	Verificar se o sistema salva corretamente as metas mensais de leitura do usuário
Localização	Teste de Requisição: POST Classe MonthGoalController <code>/monthgoals</code>
Pré-condições	Usuário autenticado e existir no banco.
Passo a passo	<ol style="list-style-type: none"> 1. Enviar requisição POST <code>/monthgoals</code> com uma meta mensal 2. Verificar o Status retornado e se a meta foi salva corretamente no banco.
Nível de teste	Integração
Técnica de teste	Caixa Preta
Critério de seleção de entradas	Baseados em casos de uso
CT04	Cadastrar e autenticar comprador
Entradas	<pre>JSON { "comprador": { "id": 1 }, "metasMensais": ["Uma Família Feliz"] }</pre>

Resultados esperados	Retorno 200(Ok) e Objeto Goal salvo e associado ao comprador.
Prioridade	Média
Pós-condições	<i>Meta mensal é salva no banco de dados e atribuída ao Comprador</i>
Resultado Obtido	<div> ✓ MonthGoalTest (com.project.pin.integra) </div> <div> ✓ CT04 - Deve salvar meta mensal vinculada ao comprador - Goal Salvo no Banco </div>
Classificação	Aprovado!

RF05	Personalização de metas de leitura diária
RT04	Integração entre GoalController e GoalRepository
Objetivo do roteiro de teste	Verificar se o sistema salva corretamente as metas de leitura diárias do usuário
Localização	Teste de Requisição POST da classe GoalController "/goals/paginas/{idcomprador}"
Pré-condições	Usuário autenticado e existir no banco.
Passo a passo	<ol style="list-style-type: none"> 1. Enviar requisição POST para "/goals/paginas/{idcomprador}" contendo o corpo com os dados de entradas. 2. Verificar o Status retornado e se a meta foi salva corretamente no banco.
Nível de teste	Integração
Técnica de teste	Caixa Preta
Critério de seleção de entradas	Baseados em casos de uso
CT05	Cadastrar e autenticar comprador
Entradas	<pre>JSON { "quantidadePaginas": 15, "diasLidos": ["Segunda", "Sexta", "Sabado"] }</pre>

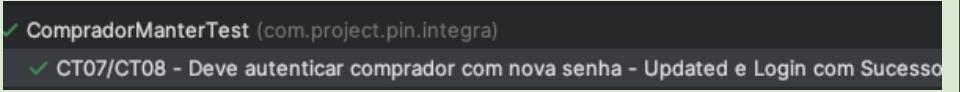
Resultados esperados	Meta de leitura diária cadastrada com sucesso e associada ao comprador.
Prioridade	Média
Pós-condições	<i>Meta diária é salva no banco de dados e atribuída ao Comprador</i>
Resultado Obtido	<div> ✓ GoalIntegrationTest (com.project.pin.integra) </div> <div> ✓ CT05 - Salvar meta diária com quantidade de páginas e dias da semana </div>
Classificação	<i>Aprovado!</i>

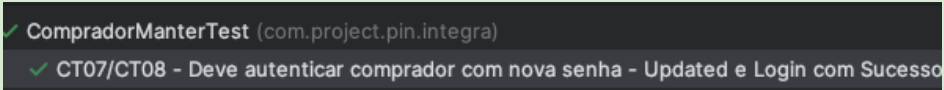
RF08	O Sistema deve possibilitar manter os livros e autores, assim como o vínculo correto entre as entidades.
RT05	Integração entre LivroController e LivroRepository
Objetivo do roteiro de teste	Verificar se o exibe corretamente o livro, o autor, verificando as relações entre entidades
Localização	Teste de Requisição POST: <code>"/livros"</code> em Classe LivroController
Pré-condições	<ol style="list-style-type: none"> 1. O banco deve estar ativo 2. O autor deve existir/estar cadastrado.
Passo a passo	<ol style="list-style-type: none"> 1. Salvar um autor válido no banco
Nível de teste	Integração
Técnica de teste	Caixa Preta
Critério de seleção de entradas	Baseados em casos de uso
CT06	Cadastro de Livro
Entradas	<pre> JSON { "nomeLivro": "Bom dia, Verônica", "isbn": "ISBN-978-6559211043", "sinopse": "Verônica Torres trabalha no Departamento de Homicídios e de Proteção à Pessoa, da Polícia Civil em São Paulo. É secretária de Carvana, um delegado pouco confiável, e filha de um respeitado policial, que teve um fim trágico e não totalmente </pre>

	<pre> esclarecido.", "pages": 320, "anoPublicado": 2022, "precoUnit": 52.40, "img": "https://m.media-amazon.com/images/I/61k-lrleRAL _SY522_.jpg", "autorId": %d * } </pre>
Resultados esperados	Retorno JSON contendo lista de livros e respectivos autores
Prioridade	Média
Pós-condições	<i>Dados exibidos corretamente na tela inicial do acervo de livros</i>
Resultado Obtido	 <pre> assertThat(response.getStatusCode()).isEqualTo(HttpStatus.CREATED); assertThat(livroRepository.count()).isEqualTo(expected: 1); Livro livro = livroRepository.findAll().get(0); assertThat(livro.getNomeLivro()).isEqualTo(expected: "Bom dia, Verônica"); assertThat(livro.getIsbn()).isEqualTo(expected: "ISBN-978-6559211043"); assertThat(livro.getPages()).isEqualTo(expected: 320); assertThat(livro.getAnoPublicado()).isEqualTo(expected: 2022); assertThat(livro.getPrecoUnit()).isEqualTo(expected: 52.40); assertThat(livro.getSinopse()).contains("Verônica Torres trabalha no Departamento de Homicídios"); assertThat(livro.getImg()).contains("m.media-amazon.com"); assertThat(livro.getAutor()).isNotNull(); assertThat(livro.getAutor().getNome()).isEqualTo(expected: "Andrea Killmore"); </pre>
Classificação	<i>Aprovado!</i>

*representa o ID do Autor salvo anteriormente como pré-condição no início da classe **integra/LivroCadastroTest** no método **setup()**.

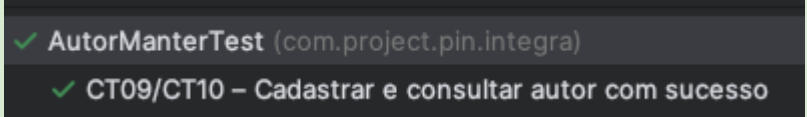
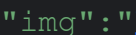
RF01	Manter Usuário (Atualizar Senha e Login)
RT06	Integração entre Comprador e Autenticação
Objetivo do roteiro de teste	Verificar se o sistema consegue atualizar os dados de senha e logar com os novos dados cadastrais.
Localização	Teste de Requisição POST: <code>"/compradores/atualizar/{id} e /auth/login"</code> na Classe <code>AuthCompradorController</code>

Pré-condições	<ol style="list-style-type: none"> 1. Banco de dados ativo. 2. Comprador Existir.
Passo a passo	<ol style="list-style-type: none"> 1. Salvar o Comprador para atualizar dados 2. Realizar a alteração dos dados da senha via PUT <code>/compradores/atualizar/{id}</code> 3. Verificar se os dados foram atualizados e se é possível realizar o novo login em <code>/auth/login</code>.
Nível de teste	Integração
Técnica de teste	Caixa Preta
Critério de seleção de entradas	Baseados em casos de uso
CT07	Atualizar comprador
Entradas	<pre>JSON { "id": %d*, "nome": "Clara Becker", "email": "clara.becker@email.com", "cpf": "123.456.789-00", "username": "clarabecker", "senha": "novaSenha123", "rua": "Rua Velha, 88", "telefone": "(48) 98888-0000", "bairro": "Centro", "cidade": "Florianópolis", "cep": "88000-000", "estado": "SC", "img": "perfil_clara.jpg", "nivel": 1, "recebeuDesconto": false }</pre>
Resultados esperados	<i>Cadastro atualizado com sucesso</i>
Prioridade	Alta
Pós-condições	<i>Dados atualizados do comprador salvos no banco de dados e próxima autenticação deverá ser com os novos dados.</i>
Resultado Obtido	<div>  </div>
Classificação	<i>Aprovado!</i>
CT08	Autenticar comprador com novos dados

Entradas	<pre>JSON { "username": "clarabecker", "password": "novaSenha123" }</pre>
Resultados esperados	<i>Retorno 200 (OK)</i>
Prioridade	<i>Alta</i>
Pós-condições	<i>Autenticado com sucesso</i>
Resultado Obtido	
Classificação	<i>Aprovado!</i>

%d é o valor recuperado do ID do Comprador salvo anteriormente como parte da pré-condição no início da classe **integra/CompradorManterTest no método **setup()**.*

RF08	Cadastrar Autor
RT07	Integração entre AutorController e AutorRepository
Objetivo do roteiro de teste	Verificar se o sistema realiza corretamente as operações de cadastro e consulta.
Localização	Teste de Requisição POST: <code>"/autores"</code> e GET: <code>"/autores/{idAutor}"</code> da Classe AutorController
Pré-condições	1. Banco de dados ativo
Passo a passo	<ol style="list-style-type: none"> 1. Enviar requisição POST para <code>"/autores"</code>. 2. Enviar JSON com dados de entrada para cadastro (CT09) 3. Validar Status 201(CREATED) e corpo da requisição contendo o Autor persistido. 4. Enviar requisição GET <code>"/autores/{idAutor}"</code> 5. Verificar Status 200(ok), contendo no corpo da requisição o nome do Autor e a lista de livros vazia.
Nível de teste	Integração
Técnica de teste	Caixa Preta
Critério de seleção de entradas	Baseados em casos de uso

CT09	Cadastrar Autor
Entradas	<pre> JSON { "nome": "Stephen Edwin King", "sobre": "Stephen Edwin King (Portland, 21 de setembro de 1947) é um escritor norte-americano de terror, ficção sobrenatural, suspense, ficção científica e fantasia. Seus livros já venderam mais de 400 milhões de cópias, com publicações em mais de 40 países. É o 9º autor mais traduzido no mundo", "img": "https://encrypted-tbn0.gstatic.com/images?q=tbn :ANd9GcR-VJNoYJFEVOWKGX5j5S07tiR_VayOp4uGviLxftr hf-W9FuajiZ4K8KpFf_g_L1F769YpOZve7zp_xH58LYeGcDu nGauKyFB_5mRqjswW" } </pre>
Resultados esperados	Retorno JSON com status 201 (Created) e Autor Salvo
Prioridade	Alta
Pós-condições	<i>Autor disponível no banco e disponível para requisição GET e disponível para cadastro de na lista de livros.</i>
Resultado Obtido	
Classificação	Aprovado!
CT10	Consultar Autores
Entradas:	GET: /autores/{idAutor}•
Resultados esperados	<pre> Status 200 (ok) JSON { "id":1, "nome":"Stephen Edwin King", "sobre":"Stephen Edwin King (Portland, 21 de setembro de 1947) é um escritor norte-americano de terror, ficção sobrenatural, suspense, ficção científica e fantasia. Seus livros já venderam mais de 400 milhões de cópias, com publicações em mais de 40 países. É o 9º autor mais traduzido no mundo",  "img":"https://encrypted-tbn0.gstatic.com/images </pre>

	?q=tbn:ANd9GcR-VJNoYJFEVOWKGX5j5S07tiR_VayOp4uGviLxfrthf-W9FuajiZ4K8KpFf_g_L1F769YpOZve7zp_xH58LYeGcDunGauKyFB_5mRqjswW", "listLivros":[]}"
Prioridade	Alta
Pós-condições	<i>O Autor existe no banco e pode ser consultado para adicionar livros.</i>
Resultado Obtido	<div> ✓ AutorManterTest (com.project.pin.integra) ✓ CT09/CT10 – Cadastrar e consultar autor com sucesso </div>
Classificação	Aprovado!

*O ID do autor é recuperado no corpo da requisição após o cadastro e utilizado para o requisição GET de consulta.

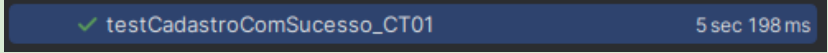
RF09 + RF10	O sistema deve manter adicionar e calcular o valor total da compra adicionada no carrinho para a finalização.
RF08	Integração entre Carrinho Controller e LivroRepository
Objetivo do roteiro de teste	Verificar a integração do sistema em adicionar corretamente os itens no carrinho e realizar o cálculo de compra corretamente com diferentes valores e quantidades.
Localização	Teste de Requisição POST: <code>"/carrinho/add"</code> para adicionar livros no carrinho e GET: <code>/carrinho/total</code> em CarrinhoTotalTest no método <code>deveAdicionarLivrosECalcularTotal()</code>
Pré-condições	<ol style="list-style-type: none"> 1. Banco de dados ativo 2. Usuário estar autenticado 3. Livro e Autor devem estar cadastrado no banco de dados
Passo a passo	<ol style="list-style-type: none"> 1. Enviar requisição POST para <code>/carrinho/add</code> com corpo JSON contendo <code>idLivro</code> e <code>quantidade</code> 2. Enviar requisição GET para <code>carrinho/total</code>.
Nível de teste	Integração
Técnica de teste	Caixa Preta
Critério de seleção de	Baseados em casos de uso

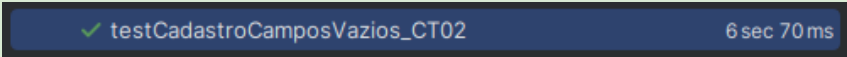
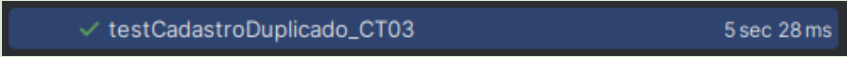
entradas	
CT11	Adicionar Livro do Carrinho
Entradas*	<p>Primeiro POST (LIVRO 1)</p> <pre>{ "idLivro": %d, "quantidade": 1 }</pre> <p>Segundo POST (LIVRO 2)</p> <pre>{ "idLivro": %d, "quantidade": 2 }</pre>
Resultados esperados	Retorno JSON com status 201 (Created) e informações do autor e do livro adicionado. Retorna a quantidade adicionada de cada livro e o valor unitário.
Prioridade	Alta
Pós-condições	<i>Lista de itens da sessão é atualizada corretamente</i>
Resultado Obtido	<div>✓ CarrinhoTotalTest (com.project.pin.integra)</div> <div>✓ CT14/CT15 – Adicionar livros e calcular total do carrinho</div>
Classificação	Aprovado!
CT12	Realizar o Cálculo para finalizar Compra do Carrinho
Entradas	GET: <code>carrinho/total</code>
Resultados esperados	Retorno 200 (ok) e corpo com o total de R\$ 169.6
Prioridade	Alta
Pós-condições	<i>O carrinho mantém os itens adicionados e o cálculo do total reflete os preços e quantidades dos livros</i>
Resultado Obtido	<div><code>assertThat(totalResp.getBody()).contains("169.6");</code></div> <div>✓ CarrinhoTotalTest (com.project.pin.integra)</div> <div>✓ CT14/CT15 – Adicionar livros e calcular total do carrinho</div>
Classificação	Aprovado!

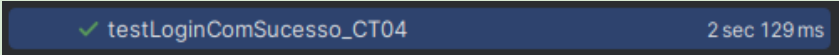
* NOTA: Será adicionado 2 vezes i.e será realizado 2 requisição para verificar a integração dos módulos (O valor dos livros cadastrado são 50.0 e 59.8, respectivamente, resultando em 169.6)

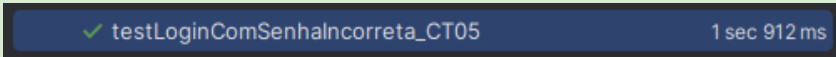
4.3 TESTES DE SISTEMA

RF01	O Sistema deve permitir ao usuário manter perfil como comprador.
RT01	Realização de cadastro do usuário no sistema
Objetivo do roteiro de teste	Verificar o cadastro de usuários e sua respectiva validação de dados
Localização	Tela de cadastro de usuário
Pré-condições	O usuário deve acessar a tela de cadastro
Passo a Passo	1. Acessar a tela de cadastro. 2. Preencher os campos com os dados correspondentes (ex: cpf, username, password). 3. Cadastrar usuário.
Nível de teste	Sistema
Técnica de teste	Caixa Preta
Critério de seleção de entradas	Baseado em casos de uso
CT01	Valida se o sistema efetua o cadastro de um usuário de forma correta
Entrada	Informar nome: "Clara Becker" Informar e-mail: "clara.becker@email.com" Informar senha: "minhaSenha123" Informar CPF: "123.456.789-00" Informar username: "clarabecker" Informar rua: "Rua das Flores, 120" Informar telefone: "(48) 99999-8888"

	<p>Informar bairro: "Centro"</p> <p>Informar cidade: "Florianópolis"</p> <p>Informar CEP: "88000-000"</p> <p>Informar estado: "SC"</p> <p>Enviar imagem (img): "perfil_clara.jpg"</p>
Resultado Esperado	<i>O sistema deve exibir uma mensagem de sucesso após o cadastro bem-sucedido.</i>
Prioridade	<i>Alta</i>
Pós-condições	<i>Usuário cadastrado no sistema</i>
Resultado Obtido	
Classificação	<i>Aprovado!</i>
CT02	Valida se o sistema impede a criação de usuários com campos vazios
Entrada	<p>Informar nome: " "</p> <p>Informar e-mail: "clara.becker@email.com"</p> <p>Informar senha: "minhaSenha123"</p> <p>Informar CPF: "123.456.789-00"</p> <p>Informar username: " "</p> <p>Informar rua: "Rua das Flores, 120"</p> <p>Informar telefone: "(48) 99999-8888"</p> <p>Informar bairro: "Centro"</p> <p>Informar cidade: "Florianópolis"</p> <p>Informar CEP: "88000-000"</p> <p>Informar estado: "SC"</p> <p>Enviar imagem (img): "perfil_clara.jpg"</p>
Resultado	<i>O sistema deve exibir uma mensagem informando que</i>

esperado	<i>todos os campos devem ser preenchidos</i>
Prioridade	<i>Média</i>
Pós-condições	<i>O usuário não é cadastrado no sistema</i>
Resultado Obtido	
Classificação	<i>Aprovado!</i>
CT03	Verifica se o sistema impede o cadastro com um username ou cpf já cadastrados
Entrada	<p>Informar nome: "Clara Becker"</p> <p>Informar e-mail: "clara.becker@email.com"</p> <p>Informar senha: "minhaSenha123"</p> <p>Informar CPF: "123.456.789-00"</p> <p>Informar username: "clarabecker"</p> <p>Informar rua: "Rua das Flores, 120"</p> <p>Informar telefone: "(48) 99999-8888"</p> <p>Informar bairro: "Centro"</p> <p>Informar cidade: "Florianópolis"</p> <p>Informar CEP: "88000-000"</p> <p>Informar estado: "SC"</p> <p>Enviar imagem (img): "perfil_clara.jpg"</p>
Resultado esperado	<i>Mensagem de erro informando que já existe um usuário cadastrado com esse username ou cpf</i>
Prioridade	<i>Alta</i>
Pós-condições	<i>Usuário não é cadastrado</i>
Resultado Obtido	
Classificação	<i>Aprovado!</i>

RF01	O Sistema deve realizar a autenticação dos usuários.
RT01	Autenticação do usuário no sistema
Objetivo do roteiro de teste	Verificar se a autenticação do usuário ocorre de forma correta
Localização	Tela de login
Pré-condições	O usuário deve acessar a tela de login
Passo a Passo	<ol style="list-style-type: none"> 1. Acessar a tela de login. 2. Preencher os campos com os dados correspondentes (username, password). 3. Autenticar usuário.
Nível de teste	Sistema
Técnica de teste	Caixa Preta
Critério de seleção de entradas	Baseado em casos de uso
CT04	Valida se o sistema efetua a autenticação de forma correta
Entrada	<ul style="list-style-type: none"> • Informar username: "clarabecker" • Informar senha: "minhaSenha123"
Resultado Esperado	<i>O sistema deve exibir uma mensagem de autenticação bem-sucedida.</i>
Prioridade	<i>Alta</i>
Pós-condições	<i>Usuário é autenticado no sistema</i>
Resultado Obtido	
Classificação	<i>Aprovado!</i>
CT05	Valida se o sistema impede a autenticação com senha incorreta
Entrada	<ul style="list-style-type: none"> • Informar username: "clarabecker" • Informar senha: "minhaSenha"

Resultado esperado	<i>O sistema deve exibir uma mensagem informando que username ou senha estão incorretos</i>
Prioridade	<i>Alta</i>
Pós-condições	<i>O usuário não é autenticado no sistema</i>
Resultado Obtido	
Classificação	<i>Aprovado!</i>

4.3 TESTES DE ACEITAÇÃO

<i>RU01</i>	O Sistema deve ser de fácil utilização e intuitivo para os usuários.
<i>RT01</i>	Cadastrar metas mensais e diárias
Objetivo do roteiro de teste	Avaliar se o sistema permite que o usuário cadastre metas de obras para leitura mensal e também metas de páginas lidas diariamente e em quais dias da semana. Verificando também a intuitividade, eficiência e validação dos dados cadastrados no sistema.
Localização	Tela de cadastro de metas
Pré-condições	O usuário deve estar autenticado no sistema.
Passo a Passo	<ol style="list-style-type: none"> 1. O usuário deve acessar a funcionalidade “Minhas metas”, para cadastro de metas 2. No componente principal o usuário deve selecionar as checkboxes dos dias de leitura desejados 3. No campo “Quantidade de páginas” o usuário deve informar um número 4. No componente “Metas Leitura Mensal” o usuário deve preencher o campo e selecionar o botão “adicionar” 5. Após feitas as operações o usuário deve selecionar a opção “salvar”
Nível de teste	Aceitação
Técnica de teste	SUS – System Usability Scale
Critério de seleção de entradas	Valores corretos, buscando averiguar efetividade, eficiência e satisfação.

CT01	Cadastrar meta de leitura mensal
Entrada	<ul style="list-style-type: none"> Informar título da obra: "A paixão segundo G.H"
Resultado esperado	<i>Mensagem de criação de meta realizada com sucesso</i>
Prioridade	<i>Média</i>
Pós-condições	<i>Meta cadastrada no sistema</i>
Perguntas para serem realizadas	<ol style="list-style-type: none"> 1. Eu achei o sistema fácil de usar para cadastrar minhas metas de leitura mensal. 2. Eu achei o processo de cadastrar as metas de leitura mensal desnecessariamente complexo. 3. Consegui encontrar as opções para cadastrar as metas de títulos mensal de forma intuitiva. 4. O sistema pareceu muito lento e travado durante o processo de cadastro das metas de leitura mensal. 5. Senti-me satisfeito(a) após conseguir completar o cadastro das metas de leitura mensal.
Resultados Obtidos	<p>Usuário 1:</p> <ol style="list-style-type: none"> 1) 5 2) 1 3) 3 4) 1 5) 5 <p>Usuário 2:</p> <ol style="list-style-type: none"> 1) 4 2) 1 3) 3 4) 1 5) 4 <p>Usuário 3:</p> <ol style="list-style-type: none"> 1) 5 2) 1 3) 5 4) 1 5) 5
CT02	Criação de meta de leitura diária
Entrada	<ul style="list-style-type: none"> Informar quantidade de páginas : "15" Selecionar 5 checkboxes
Resultado	<i>Mensagem de criação de meta realizada com sucesso</i>

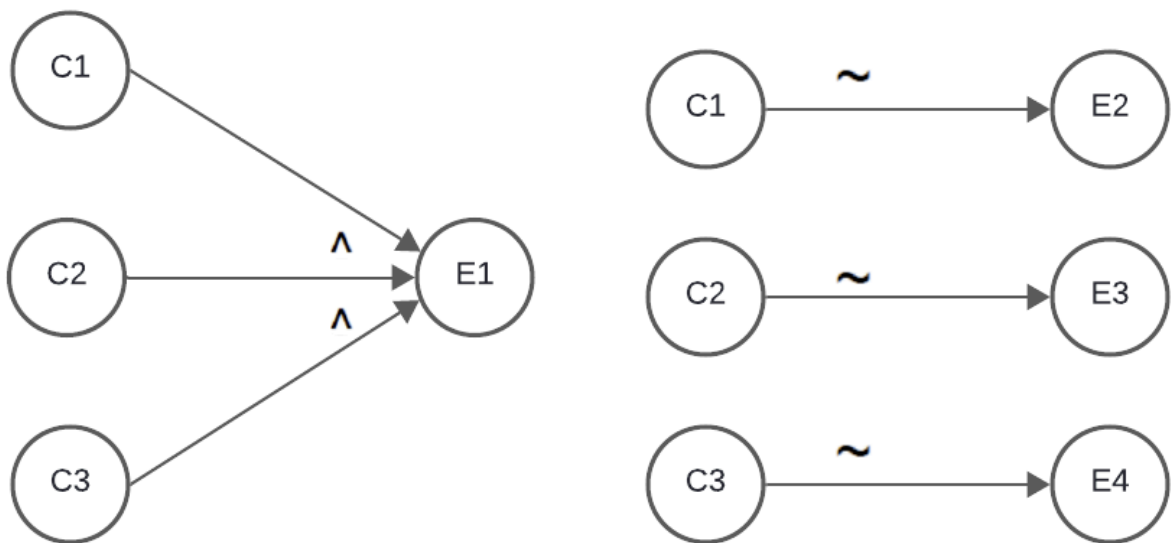
esperado	
Prioridade	<i>Média</i>
Pós-condições	<i>Meta cadastrada no sistema</i>
Perguntas para serem realizadas	<ol style="list-style-type: none"> 1. Eu achei o sistema fácil de usar para cadastrar minhas metas de páginas diárias e dias da semana. 2. Eu achei o processo de cadastrar as metas de leitura diária desnecessariamente complexo. 3. Consegui encontrar as opções para cadastrar as metas de leitura diária de forma intuitiva. 4. O sistema pareceu muito lento e "travado" durante o processo de cadastro das metas de leitura diárias. 5. Senti-me satisfeito(a) após conseguir completar o cadastro das metas de leitura diária.
Resultado Obtido	<p>Usuário 1:</p> <p>6) 5 7) 1 8) 5 9) 1 10) 5</p> <p>Usuário 2:</p> <p>6) 4 7) 2 8) 3 9) 1 10) 4</p> <p>Usuário 3:</p> <p>6) 4 7) 1 8) 4 9) 1 10) 5</p>

Resultado da escala SUS por avaliação de Usuário		
Usuário	Pontuação	Resultado
Usuário 1	95	Excelente
Usuário 2	77,5	Boa Usabilidade
Usuário 3	95	Excelente

5. GRAFOS CAUSA E EFEITO

Unitário RF02

	R1	R2	R3	R4	R5
C1: Usuário preenche username corretamente	V	V	V	F	F
C2: Usuário preenche senha corretamente	V	V	F	V	F
C3: Usuário Cadastrado	V	F	V	V	V
E1: Usuário é autenticado com sucesso	V	F	F	F	F
E2: Username Inválido	F	V	F	V	V
E3: Senha Inválida	F	V	V	F	V
E4: Usuário não cadastrado	F	V	F	F	F



Unitário RF08

	R1	R2	R3	R4	R5
C1: Autor existe no sistema	V	V	V	F	F
C2: Livro existe no sistema	V	F	V	F	V
C3: Autor está vinculado ao livro correspondente	V	F	F	F	F*

E1: Livro e Autor encontrados com sucesso	V	F	F	F	F
E2: Autor não cadastrado	F	F	F	V	V
E3: Livro não existe no sistema	F	V	F	V	F
E4: Vínculo de livro e autor incorreto	F	F	V	F	F*

* **Nota:** Se autor nem existe não podemos afirmar Verdadeiramente (V) que o vínculo está incorreto.

