

Relatório Final de Testes – Econstore.

Disciplina: Teste de Software – quarta feira matutino

Professor: Oscar Galdino

Nome dos integrante: Gustavo Medeiros, Gabriel Sampaio, Kalebe Lucas, Gustavo Fernandes, Pablo Gabriel

1. Introdução ao Sistema Desenvolvido

Este relatório apresenta os resultados dos testes de software realizados no sistema Econstore, uma plataforma de e-commerce que permite a compra e venda de produtos. O sistema é composto por um frontend web e um backend que interage com um banco de dados MySQL. O objetivo principal da Econstore é proporcionar uma plataforma para que clientes possam navegar por produtos, realizar cadastros de conta, adicionar itens ao carrinho de compras e finalizar pedidos. O sistema também oferece funcionalidades administrativas para lojistas, permitindo o gerenciamento de produtos e o acompanhamento de pedidos realizados.

2. Planejamento de Testes

Objetivo dos Testes

O objetivo dos testes no sistema Econstore é garantir que todas as funcionalidades essenciais estejam implementadas corretamente e que o sistema se comporte conforme os requisitos definidos, tanto para os perfis de cliente quanto de lojista (funcionário). Além disso, os testes visam identificar falhas ou comportamentos indesejados, validar os fluxos de compra e gerenciamento de produtos, e assegurar a integridade dos dados.

Escopo do Teste

Funcionalidades que serão testadas:

- Login e cadastro de clientes.
- Login de funcionários.
- Adição, remoção e visualização de produtos no carrinho.
- Finalização de compras com e sem dados de pagamento.
- Redução do estoque após compras.
- Criação, edição e exclusão de produtos pelo funcionário.
- Visualização de pedidos no painel administrativo.

Funcionalidades que não serão testadas:

- Responsividade em diferentes dispositivos (testes mobile).
- Performance em alta carga de usuários simultâneos.
- Funcionalidades não implementadas, como chat, avaliação de produtos ou envio de e-mails automáticos.

Ferramentas que Serão Utilizadas

- **Navegador (Chrome)** – para testes manuais das funcionalidades web.
- **MySQL Workbench** – para consulta e verificação dos dados no banco de dados.
- **Console do navegador / DevTools** – para inspecionar erros de JavaScript, rede e comportamento da interface.
- **VS Code** – para leitura e inspeção do código-fonte durante os testes unitários.
- **Robot Framework** – para automatizar testes E2E.
- **Jest** - para testes unitários de backend.
- **Postman** – testes de API.

Estratégia de Testes

Tipos de testes aplicados:

- **Testes Unitários:** A estratégia de testes unitários envolveu a simulação da camada de banco de dados (MySQL) através de mocks, permitindo que os testes fossem rápidos, confiáveis e independentes de um ambiente de banco de dados real.

As funções testadas são responsáveis por diversas operações essenciais, incluindo: Criação, edição e listagem de produtos (RF1). Atualização e controle de estoque (RF2). Cadastro e login de clientes e funcionários (RF4). Validação de entrada de dados e simulação de falhas (como emails/CPFs duplicados, RN02, RN06).

- **Testes End-to-End (E2E):** Simulação do fluxo completo de um usuário, como uma compra desde o login até a finalização.
- **Postman(API):** Testes de interação da API com requisições HTTP.

Critérios de Aceitação:

- O sistema deve exibir mensagens claras em casos de erro ou sucesso.
- Os fluxos devem seguir o esperado sem falhas ou interrupções.
- O banco de dados deve refletir corretamente todas as ações realizadas.

Critérios de saída do ciclo de testes:

- Todos os testes previstos foram executados.
- Todos os defeitos identificados foram documentados com justificativa.
- O sistema está funcionando conforme os requisitos mínimos definidos.

Papéis e Responsabilidades Dentro do Grupo

Nome do Integrante	Responsabilidade Principal
[Gustavo]	Criação dos casos de teste e Execução dos testes de fluxo completo (E2E)
[Kalebe]	Testes unitários no backend via Jest
[Gabriel]	Documentação dos resultados dos testes e organização do relatório final

3. Descrição dos Casos de Teste

A seguir, são detalhados os casos de teste elaborados para o sistema Econstore, seguindo o formato especificado para facilitar a visualização e o acompanhamento. Os casos de teste estão organizados por funcionalidade para uma melhor compreensão. Os testes incluem:

- **Testes unitários:** Validação de funções e classes individuais do código-fonte.
- **Testes E2E (End-to-End):** Testes que simulam a interação completa do usuário com o sistema, desde o início ao fim de um fluxo de trabalho.
- **Testes de API:** Testes para validar as requisições HTTP.

TESTES UNITÁRIOS:

Foram elaborados para validar o comportamento esperado de cada método nos modelos e controladores do backend, conforme as Regras de Negócio (RNs) e Requisitos Funcionais (RFs) do projeto. No total, foram implementados e executados 83 testes unitários.

ProductModel.js

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Criar produto com sucesso	Unitário	Retornar ID e nome do produto	RF1, RN1
Lançar erro ao falhar na criação	Unitário	Retornar erro específico	
Atualizar estoque com valor válido	Unitário	Diminuir quantidade corretamente	RF2, RN04
Lançar erro se quantidade solicitada > estoque (RN04)	Unitário	Retornar mensagem "Estoque insuficiente"	RF2, RN04
Atualizar estoque com conexão fornecida	Unitário	Diminuir quantidade sem liberar conexão	
Atualizar estoque sem conexão fornecida	Unitário	Diminuir quantidade e liberar conexão	
Retornar todos os produtos sem filtros	Unitário	Retornar uma lista completa de produtos	RF1
Retornar produtos filtrados por categoria	Unitário	Retornar lista de produtos da categoria especificada	RF1
Retornar produtos filtrados por nome	Unitário	Retornar lista de produtos com nome correspondente	RF1
Retornar array vazio se nenhum produto for encontrado	Unitário	Retornar um array vazio	RF1
Lançar erro em caso de falha genérica do BD	Unitário	Lançar erro de banco de dados	
Retornar produto existente por ID	Unitário	Retornar os dados do produto	RF1
Retornar undefined se produto não encontrado por ID	Unitário	Retornar undefined	RF1
Lançar erro em getProductById	Unitário	Lançar erro de banco de dados	
Atualizar produto com sucesso	Unitário	Retornar true	RF1, RN1
Retornar false se produto não encontrado p/ atualização	Unitário	Retornar false	
Lançar erro em updateProduct	Unitário	Lançar erro de banco de dados	

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Deletar produto com sucesso	Unitário	Retornar true	RF1, RN1
Retornar false se produto não encontrado p/ exclusão	Unitário	Retornar false	
Lançar erro se produto associado a pedido	Unitário	Lançar erro "associado a um ou mais pedidos"	
Lançar erro em deleteProduct	Unitário	Lançar erro de banco de dados	

UserModel.js

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Criar usuário com sucesso	Unitário	Retornar ID e e-mail do usuário com senha hashed	RF4, RN2
Lançar erro se e-mail já cadastrado (RN06)	Unitário	Lançar erro "E-mail já cadastrado"	RF4, RN06
Lançar erro se CPF já cadastrado (RN02)	Unitário	Lançar erro "CPF já cadastrado"	RF4, RN2
Lançar erro em caso de falha genérica do BD	Unitário	Lançar erro do banco de dados	
Retornar usuário existente por e-mail	Unitário	Retornar os dados do usuário	RF4
Retornar undefined se usuário não encontrado p/ e-mail	Unitário	Retornar undefined	RF4
Lançar erro em findUserByEmail	Unitário	Lançar erro de banco de dados	
Retornar usuário existente por ID	Unitário	Retornar os dados do usuário	RF4
Retornar undefined se usuário não encontrado p/ ID	Unitário	Retornar undefined	RF4
Lançar erro em findUserById	Unitário	Lançar erro de banco de dados	

AuthController.test.js

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Registrar cliente com dados válidos	Unitário	Retornar status 201 e dados do usuário	RF4, RN2
Registrar cliente sem campos obrigatórios	Unitário	Retornar status 400 e mensagem de erro	RF4, RN2
Registrar cliente com email duplicado (RN06)	Unitário	Retornar status 409 e mensagem "Email já cadastrado"	RF4, RN06
Registrar cliente com CPF duplicado (RN02)	Unitário	Retornar status 409 e mensagem "CPF já cadastrado"	RF4, RN2
Registrar cliente com erro interno do servidor	Unitário	Retornar status 500 e mensagem de erro genérico	RF4
Login de cliente sem preencher campos	Unitário	Retornar status 400 e mensagem "campos obrigatórios"	
Login de cliente com dados inválidos	Unitário	Retornar status 401 para credenciais inválidas	RF4, RN03
Login de cliente com dados válidos	Unitário	Retornar status 200 e token JWT	RF4, RN03
Login de cliente com erro interno do servidor	Unitário	Retornar status 500 e mensagem de erro genérico	

ProductController.test.js

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Criar produto com sucesso	Unitário	Retornar status 201 e dados do produto	RF1, RN1
Criar produto sem campos obrigatórios	Unitário	Retornar status 400 e mensagem de erro	RF1, RN1
Criar produto com erro interno do servidor	Unitário	Retornar status 500 e mensagem de erro genérico	
Retornar todos os produtos com sucesso	Unitário	Retornar status 200 e lista de produtos	RF1
Retornar erro interno ao buscar todos produtos	Unitário	Retornar status 500 e mensagem de erro genérico	
Retornar produto existente por ID	Unitário	Retornar status 200 e dados do produto	RF1
Retornar status 404 se produto não encontrado	Unitário	Retornar status 404 e mensagem "Produto não encontrado"	RF1
Retornar erro interno ao buscar produto por ID	Unitário	Retornar status 500 e mensagem de erro genérico	
Atualizar produto com sucesso	Unitário	Retornar status 200 e mensagem de sucesso	RF1, RN1
Retornar status 404 se produto não encontrado p/ atualização	Unitário	Retornar status 404 e mensagem "Produto não encontrado"	
Retornar status 200 se nenhuma alteração detectada	Unitário	Retornar status 200 e mensagem "Nenhuma alteração..."	
Retornar erro interno ao atualizar produto	Unitário	Retornar status 500 e mensagem de erro genérico	
Deletar produto com sucesso	Unitário	Retornar status 200 e mensagem de sucesso	RF1, RN1
Retornar status 404 se produto não encontrado p/ exclusão	Unitário	Retornar status 404 e mensagem "Produto não encontrado"	

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Retornar status 400 se produto associado a pedido	Unitário	Retornar status 400 e mensagem "Não é possível excluir..."	
Retornar erro interno ao deletar produto	Unitário	Retornar status 500 e mensagem de erro genérico	

PedidoService.js

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Criar pedido com sucesso	Unitário	Criar pedido, atualizar estoque, comitar transação	RF3, RF5, RN03, RN04
Lançar erro e fazer rollback se produto não encontrado	Unitário	Lançar erro "Produto com ID X não encontrado"	RF3, RN04
Lançar erro e fazer rollback se estoque insuficiente	Unitário	Lançar erro "Estoque insuficiente...Transação cancelada"	RF3, RN04
Lançar erro e fazer rollback se inserção pedido falhar	Unitário	Lançar erro "Falha na inserção do pedido..."	RF3
Lançar erro e fazer rollback se atualização estoque falhar	Unitário	Lançar erro "Erro ao atualizar estoque...Transação cancelada"	RF3
Lançar erro e fazer rollback se inserção item falhar	Unitário	Lançar erro "Falha ao inserir item...Transação cancelada"	RF3

PedidoController.js

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Criar pedido com sucesso	Unitário	Retornar status 201 e dados do pedido	RF3, RF5, RN03
Retornar status 500 em caso de erro no serviço	Unitário	Retornar status 500 e mensagem de erro	RF3
Retornar status 400 se faltarem produtos	Unitário	Retornar status 400 e mensagem "obrigatórios"	RF3
Retornar status 400 se faltar total	Unitário	Retornar status 400 e mensagem "obrigatórios"	RF3
Retornar status 400 se faltar status	Unitário	Retornar status 400 e mensagem "obrigatórios"	RF3
Retornar status 401 se usuário não autenticado	Unitário	Retornar status 401 e mensagem "Usuário não autenticado"	RF4

FuncionarioService.js

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Retornar token e dados do lojista p/ login	Unitário	Retornar token e objeto do usuário	RF4, RN03
Retornar null se e-mail lojista não encontrado	Unitário	Retornar null	RF4
Retornar null se senha incorreta	Unitário	Retornar null	RF4
Lançar erro em caso de falha genérica do BD	Unitário	Lançar erro de banco de dados	

VerPedidosService.js

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Retornar todos os pedidos com seus itens e dados usuário	Unitário	Retornar lista de pedidos com itens e user	RF3
Retornar pedidos c/ lista vazia de itens	Unitário	Retornar pedidos com propriedade 'itens' vazia	RF3
Retornar array vazio se nenhum pedido encontrado	Unitário	Retornar array vazio	RF3
Lançar erro em caso de falha genérica do BD	Unitário	Lançar erro de banco de dados	

VerPedidosController.js

Caso de Teste	Tipo	Resultado Esperado	Requisitos Cobertos
Retornar todos os pedidos com sucesso	Unitário	Retornar status 200 e lista de pedidos	RF3
Retornar status 500 em caso de erro no serviço	Unitário	Retornar status 500 e mensagem de erro	

TESTES E2E:

Testes de Login e Cadastro de Cliente

CT001 - Testar login de cliente sem preencher os campos

- **ID do CT:** CT001
- **Descrição:** Testar login de cliente sem preencher os campos
- **Pré-condições:** O usuário está na tela de login de cliente.
- **Passos para execução:**
 1. Acessar a tela de login de cliente.
 2. Clicar no botão “Entrar” sem preencher os campos de e-mail e senha.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Campos de e-mail e senha vazios.

- **Saída Esperada:** O sistema deve exibir uma mensagem de erro indicando que os campos de e-mail e senha são obrigatórios e o login não deve ser realizado.
- **Critério de sucesso:** O sistema exibe a mensagem de erro correta e o login não é efetuado.

CT002 - Testar login de cliente com dados inválidos

- **ID do CT:** CT002
- **Descrição:** Testar login de cliente com dados inválidos
- **Pré-condições:** O usuário está na tela de login de cliente.
- **Passos para execução:**
 1. Acessar a tela de login de cliente.
 2. Preencher o campo de e-mail com um e-mail inválido (ex: emailinvalido.com).
 3. Preencher o campo de senha com uma senha inválida.
 4. Clicar no botão “Entrar”.
- **Dados de entrada e saída esperada:**
 - **Entrada:** E-mail inválido e/ou senha inválida.
 - **Saída Esperada:** O sistema deve exibir uma mensagem de erro indicando que as credenciais são inválidas e o login não deve ser realizado.
- **Critério de sucesso:** O sistema exibe a mensagem de erro correta e o login não é efetuado.

CT003 - Testar cadastro de cliente

- **ID do CT:** CT003
- **Descrição:** Testar cadastro de cliente
- **Pré-condições:** O usuário está na tela de cadastro de cliente.
- **Passos para execução:**
 1. Acessar a tela de cadastro de cliente.
 2. Preencher todos os campos obrigatórios (nome completo, CPF, telefone, email, senha, endereço completo) com dados válidos e únicos.
 3. Clicar no botão “Cadastrar”.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Dados válidos e únicos para cadastro.
 - **Saída Esperada:** O sistema deve criar o novo usuário com sucesso, redirecionar para a tela de login, e o usuário deve ser capaz de fazer login com as credenciais cadastradas.
- **Critério de sucesso:** O cadastro é realizado com sucesso e o usuário pode fazer login.

CT004 - Testar login de cliente com dados válidos

- **ID do CT:** CT004
- **Descrição:** Testar login de cliente com dados válidos
- **Pré-condições:** O usuário está na tela de login de cliente. Existe um cliente cadastrado com e-mail e senha válidos.
- **Passos para execução:**
 1. Acessar a tela de login de cliente.
 2. Preencher o campo de e-mail com um e-mail de cliente válido.
 3. Preencher o campo de senha com a senha correspondente.
 4. Clicar no botão “Entrar”.
- **Dados de entrada e saída esperada:**
 - **Entrada:** E-mail e senha de um cliente válido.
 - **Saída Esperada:** O sistema deve autenticar o usuário com sucesso e redirecioná-lo para a página inicial do cliente.
- **Critério de sucesso:** O login é efetuado com sucesso e o usuário é redirecionado para a página correta.

CT005 - Testar botão de logout

- **ID do CT:** CT005
- **Descrição:** Testar botão de logout
- **Pré-condições:** O usuário está logado como cliente ou lojista.
- **Passos para execução:**
 1. Fazer login como cliente.
 2. Localizar e clicar no botão “Logout”.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Clique no botão de logout.
 - **Saída Esperada:** O sistema deve encerrar a sessão do usuário e redirecioná-lo para a tela de login ou página inicial do site.
- **Critério de sucesso:** A sessão é encerrada e o usuário é redirecionado para a tela de login.

Testes de Carrinho de Compras e Finalização de Compra

CT006 - Testar comprar um produto com quantidade maior que a disponível

- **ID do CT:** CT006
- **Descrição:** Testar comprar um produto com quantidade maior que a disponível
- **Pré-condições:** O usuário está logado. Existe um produto com quantidade em estoque > 0.
- **Passos para execução:**
 1. Adicionar um produto ao carrinho.

2. No carrinho, tentar aumentar a quantidade do produto para um valor maior que o estoque disponível.
 3. Finalizar a compra
- **Dados de entrada e saída esperada:**
 - **Entrada:** Tentativa de comprar quantidade maior que a disponível.
 - **Saída Esperada:** O sistema deve impedir a compra exibindo uma mensagem de erro que a quantidade solicitada é maior que a disponível.
 - **Critério de sucesso:** O sistema impede a compra e exibe a mensagem de erro correta.

CT007 - Testar remover produto do carrinho

- **ID do CT:** CT007
- **Descrição:** Testar remover produto do carrinho
- **Pré-condições:** O usuário está logado.
- **Passos para execução:**
 1. Adicionar um ou mais produtos ao carrinho.
 2. Acessar a página do carrinho de compras.
 3. Clicar no botão de remover ao lado do produto.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Clique no botão de remover produto.
 - **Saída Esperada:** O produto selecionado deve ser removido do carrinho, o valor total do carrinho deve ser atualizado, e o sistema deve exibir uma mensagem de carrinho vazio.
- **Critério de sucesso:** O produto é removido, o carrinho é atualizado e a mensagem de confirmação é exibida.

CT008 - Testar comprar produto sem preencher os dados do cartão (pedido ficará com status de pendente)

- **ID do CT:** CT008
- **Descrição:** Testar comprar produto sem preencher os dados do cartão (pedido ficará com status de pendente)
- **Pré-condições:** O usuário está logado. O carrinho de compras contém produtos.
- **Passos para execução:**
 1. Prosseguir para a tela de finalização da compra.
 2. Não preencher os dados do cartão de crédito.
 3. Confirmar a compra.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Confirmação da compra sem dados do cartão.

- **Saída Esperada:** O pedido deve ser criado com sucesso, e seu status deve ser “Pendente”. O sistema deve exibir uma mensagem informando que o pedido está aguardando o pagamento.
- **Critério de sucesso:** O pedido é criado com status “Pendente” e a mensagem de aguardando pagamento é exibida.

CT009 - Testar comprar produto preenchendo os dados do cartão (pedido ficará como aprovado)

- **ID do CT:** CT009
- **Descrição:** Testar comprar produto preenchendo os dados do cartão (pedido ficará como aprovado)
- **Pré-condições:** O usuário está logado. O carrinho de compras contém produtos.
- **Passos para execução:**
 1. Prosseguir para a tela de finalização da compra.
 2. Preencher todos os dados do cartão de crédito com informações.
 3. Confirmar a compra.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Dados do cartão de crédito.
 - **Saída Esperada:** O pedido deve ser criado com sucesso, e seu status deve ser “Aprovado”. O sistema deve exibir uma mensagem de confirmação da compra.
- **Critério de sucesso:** O pedido é criado com status “Aprovado” e a mensagem de confirmação é exibida.

CT010 - Testar ver se a quantidade de produtos disponíveis diminui após a compra

- **ID do CT:** CT010
- **Descrição:** Testar ver se a quantidade de produtos disponíveis diminui após a compra
- **Pré-condições:** O usuário está logado. Existe um produto com quantidade em estoque > 0.
- **Passos para execução:**
 1. Verificar a quantidade em estoque de um produto específico.
 2. Comprar uma unidade desse produto.
 3. Após a confirmação da compra, verificar novamente a quantidade em estoque do produto.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Compra de uma unidade do produto.
 - **Saída Esperada:** A quantidade em estoque do produto deve ter diminuído em uma unidade em relação à quantidade inicial.
- **Critério de sucesso:** A quantidade em estoque é atualizada corretamente.

Testes de Login e Gerenciamento de Funcionário

CT011 - Testar login de funcionário sem preencher os campos

- **ID do CT:** CT011
- **Descrição:** Testar login de funcionário sem preencher os campos
- **Pré-condições:** O usuário está na tela de login de funcionário.
- **Passos para execução:**
 1. Acessar a tela de login de funcionário.
 2. Clicar no botão “Entrar” sem preencher os campos de e-mail e senha.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Campos de e-mail e senha vazios.
 - **Saída Esperada:** O sistema deve exibir uma mensagem de erro indicando que os campos de e-mail e senha são obrigatórios e o login não deve ser realizado.
- **Critério de sucesso:** O sistema exibe a mensagem de erro correta e o login não é efetuado.

CT012 - Testar login de funcionário com os dados inválidos

- **ID do CT:** CT012
- **Descrição:** Testar login de funcionário com os dados inválidos
- **Pré-condições:** O usuário está na tela de login de funcionário.
- **Passos para execução:**
 1. Acessar a tela de login de funcionário.
 2. Preencher o campo de e-mail com um e-mail de funcionário inválido (ex: funcionarioinvalido.com).
 3. Preencher o campo de senha com uma senha inválida.
 4. Clicar no botão “Entrar”.
- **Dados de entrada e saída esperada:**
 - **Entrada:** E-mail inválido e/ou senha inválida.
 - **Saída Esperada:** O sistema deve exibir uma mensagem de erro indicando que as credenciais são inválidas e o login não deve ser realizado.
- **Critério de sucesso:** O sistema exibe a mensagem de erro correta e o login não é efetuado.

CT013 - Testar login de funcionário com dados válidos

- **ID do CT:** CT013
- **Descrição:** Testar login de funcionário com dados válidos

- **Pré-condições:** O usuário está na tela de login de funcionário. Existe um funcionário cadastrado com e-mail e senha válidos.
- **Passos para execução:**
 1. Acessar a tela de login de funcionário.
 2. Preencher o campo de e-mail com um e-mail de funcionário válido.
 3. Preencher o campo de senha com a senha correspondente.
 4. Clicar no botão “Entrar”.
- **Dados de entrada e saída esperada:**
 - **Entrada:** E-mail e senha de um funcionário válido.
 - **Saída Esperada:** O sistema deve autenticar o funcionário com sucesso e redirecioná-lo para o painel de administração.
- **Critério de sucesso:** O login é efetuado com sucesso e o funcionário é redirecionado para a página correta.

CT014 - Testar botão de ver produtos

- **ID do CT:** CT014
- **Descrição:** Testar botão de ver produtos
- **Pré-condições:** O funcionário está logado no painel administrativo.
- **Passos para execução:**
 1. No painel administrativo, localizar e clicar no botão “Ver Produtos”.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Clique no botão “Ver Produtos”.
 - **Saída Esperada:** O sistema deve redirecionar o funcionário para a página que lista todos os produtos cadastrados no sistema.
- **Critério de sucesso:** O funcionário é redirecionado para a lista de produtos.

CT015 - Testar editar mudar todo um produto

- **ID do CT:** CT015
- **Descrição:** Testar editar mudar todo um produto
- **Pré-condições:** O funcionário está logado no painel administrativo. Existe pelo menos um produto cadastrado.
- **Passos para execução:**
 1. Localizar e clicar no botão “Ver Produtos”.
 2. Selecionar um produto para edição.
 3. Alterar todos os campos editáveis do produto (nome, descrição, preço, quantidade, imagem).
 4. clicar no botão “Salvar”.
- **Dados de entrada e saída esperada:**

- **Entrada:** Novas informações para todos os campos do produto.
- **Saída Esperada:** O sistema deve atualizar todas as informações do produto com sucesso, e as novas informações devem ser refletidas na lista de produtos e na página de detalhes do produto.
- **Critério de sucesso:** O produto é editado com sucesso e as alterações são refletidas.

CT016 - Testar excluir um produto que tem um pedido associado a ele

- **ID do CT:** CT016
- **Descrição:** Testar excluir um produto que tem um pedido dele
- **Pré-condições:** O funcionário está logado no painel administrativo. Existe um produto com pelo menos um pedido associado a ele.
- **Passos para execução:**
 1. Acessar a lista de produtos.
 2. Tentar excluir um produto que possui pedidos.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Tentativa de exclusão de produto com pedidos associados.
 - **Saída Esperada:** O sistema deve impedir a exclusão do produto e exibir uma mensagem de erro ou aviso, informando que o produto não pode ser excluído por ter pedidos associados.
- **Critério de sucesso:** O sistema impede a exclusão e exibe a mensagem de erro correta.

CT017 - Testar excluir um produto sem pedidos

- **ID do CT:** CT017
- **Descrição:** Testar excluir um produto sem pedidos
- **Pré-condições:** O funcionário está logado no painel administrativo. Existe um produto sem pedidos associados.
- **Passos para execução:**
 1. Acessar a lista de produtos.
 2. Selecionar um produto que não possui pedidos associados.
 3. Clicar no botão de exclusão.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Confirmação de exclusão de produto sem pedidos associados.
 - **Saída Esperada:** O produto deve ser removido com sucesso da lista de produtos.
- **Critério de sucesso:** O produto é excluído com sucesso.

CT018 - Testar criar um produto com dados completos

- **ID do CT:** CT018
- **Descrição:** Testar criar um produto com dados completos
- **Pré-condições:** O funcionário está logado no painel administrativo.
- **Passos para execução:**
 1. Acessar a lista de produtos.
 2. Preencher todos os campos obrigatórios (nome, descrição, preço, quantidade em estoque, URL da imagem) com dados válidos.
 3. Salvar o novo produto.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Dados completos e válidos para o novo produto.
 - **Saída Esperada:** O novo produto deve ser criado com sucesso e aparecer na lista de produtos disponíveis no sistema.
- **Critério de sucesso:** O produto é criado com sucesso e aparece na lista.

CT019 - Testar comprar o produto criado

- **ID do CT:** CT019
- **Descrição:** Testar comprar o produto criado
- **Pré-condições:** Um novo produto foi criado e está disponível para compra.
- **Passos para execução:**
 1. Acessar a página do produto recém-criado como cliente.
 2. Adicionar o produto ao carrinho.
 3. Finalizar a compra.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Compra do produto recém-criado.
 - **Saída Esperada:** A compra deve ser processada com sucesso, e o estoque do produto deve ser atualizado.
- **Critério de sucesso:** O produto criado é comprado com sucesso.

CT020 - Testar botão de ver pedidos

- **ID do CT:** CT020
- **Descrição:** Testar botão de ver pedidos
- **Pré-condições:** O funcionário está logado no painel administrativo.
- **Passos para execução:**
 1. No painel administrativo, localizar e clicar no botão “Ver Pedidos”.
- **Dados de entrada e saída esperada:**
 - **Entrada:** Clique no botão “Ver Pedidos”.
 - **Saída Esperada:** O sistema deve redirecionar o funcionário para a página que lista todos os pedidos realizados no sistema.

- **Critério de sucesso:** O funcionário é direcionado para a lista de pedidos feitos.

4. Resultados das Execuções e Evidências

Nome do caso de teste: CT 001 - Testar login de cliente sem preencher os campos

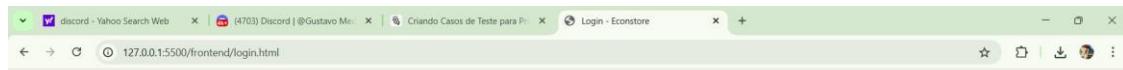
Data: 22/06/2025

Funcionalidade: Login

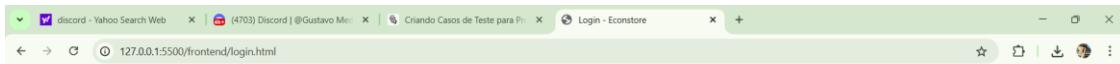
Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

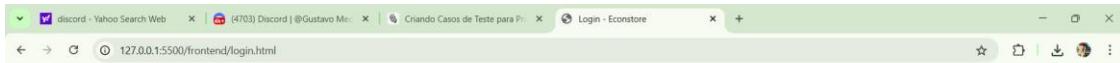
01 -

A screenshot of the 'Login Cliente' form. Both the 'Email:' and 'Senha:' fields are empty. An orange error message 'Preencha este campo.' (Fill this field.) is displayed above each field. The 'ENTRAR' button is visible below the fields.

02 -



03 –



Nome do caso de teste: CT 002 - Testar login de cliente com dados inválidos

Data: 22/06/2025

Funcionalidade: Login

Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01-

Nome do caso de teste: CT 003 - Testar cadastro de cliente

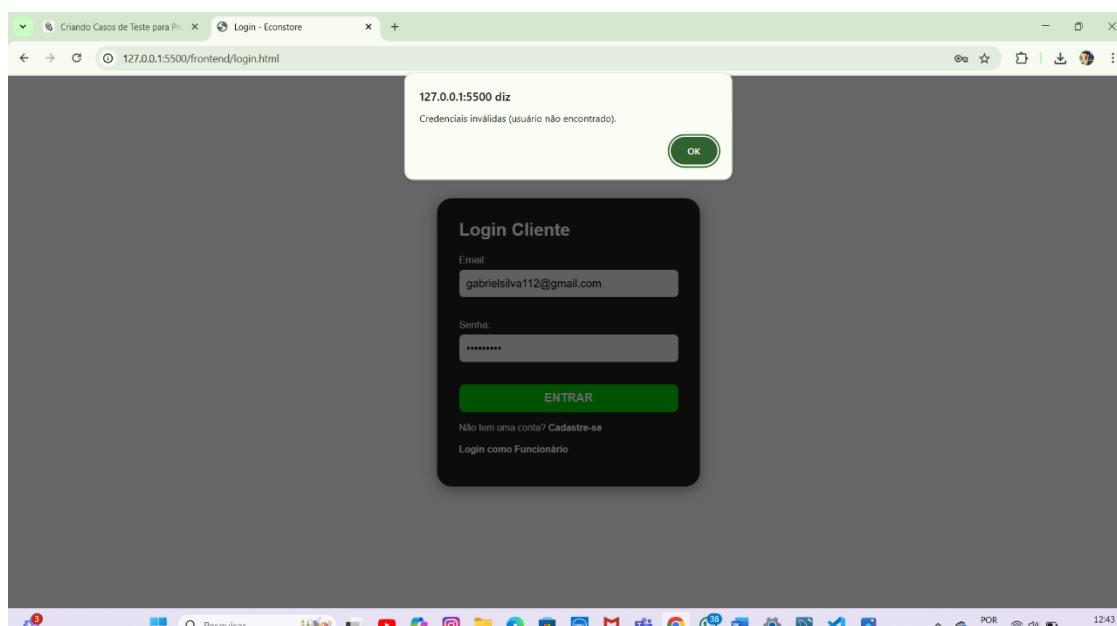
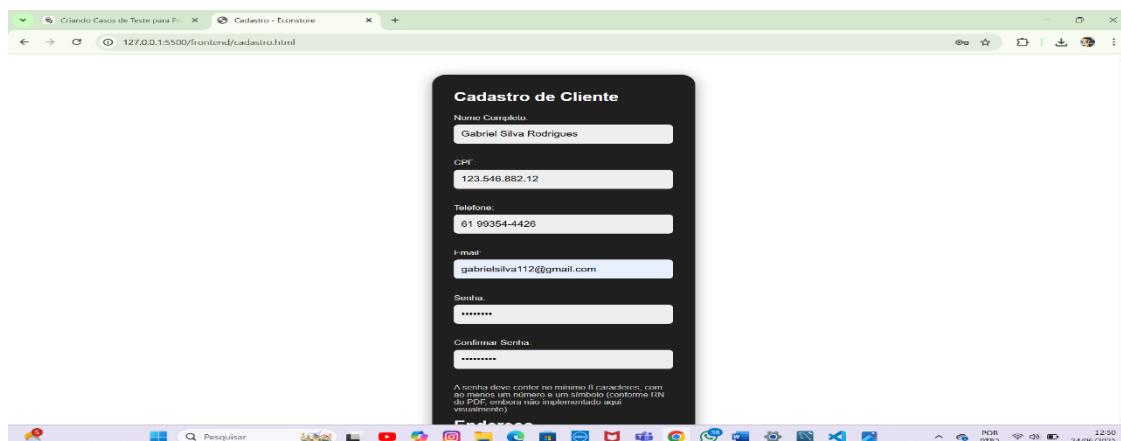
Data: 22/06/2025

Funcionalidade: Cadastro

Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01-



01 -

The screenshot shows a web browser window with the URL 127.0.0.1:5500/frontend/cadastro.html. The page title is "Criando Casos de Teste para Python". The main content is a form titled "Endereço" (Address) with the following fields:

- CEP: 75648829
- Rua: Augusta
- Número: 3
- Complemento: perto do hospital
- Bairro: 5
- Cidade: Brasília
- Estado (UF): DF

A green "CADASTRAR" button is at the bottom. Below the form, there's a link "Já tem uma conta? Faça Login". The browser taskbar shows various pinned icons and the system tray indicates the date as 24/06/2025.

02-

The screenshot shows a web browser window with the URL 127.0.0.1:5500/frontend/login.html. The page title is "Criando Casos de Teste para Python". The main content is a "Login Cliente" form with the following fields:

- Email: (empty input field)
- Senha: (empty input field)

Below the form are links: "ENTRAR", "Não tem uma conta? Cadastre-se", and "Login como Funcionário". A secondary window titled "Salvar senha?" (Save password?) is overlaid on the right side. It contains fields for "Nome de usuário" (gabriel.silva112@gmail.com) and "Senha" (redacted), and two buttons: "Salvar" (Save) and "Nunca" (Never). A note below says: "Você pode usar as senhas salvas em qualquer dispositivo. Elas ficam armazenadas no Gerenciador de senhas do Google da conta gabriel.sampaio559@gmail.com.". The browser taskbar shows various pinned icons and the system tray indicates the date as 24/06/2025.

Nome do caso de teste: CT 004 - Testar login de cliente com dados válidos

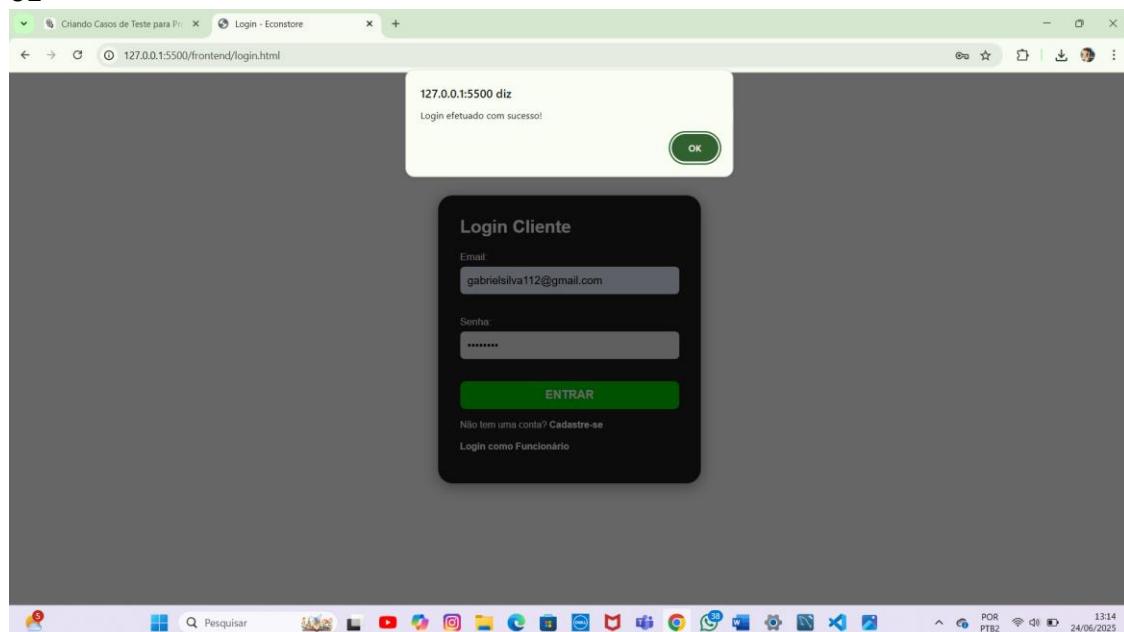
Data: 22/06/2025

Funcionalidade: Login

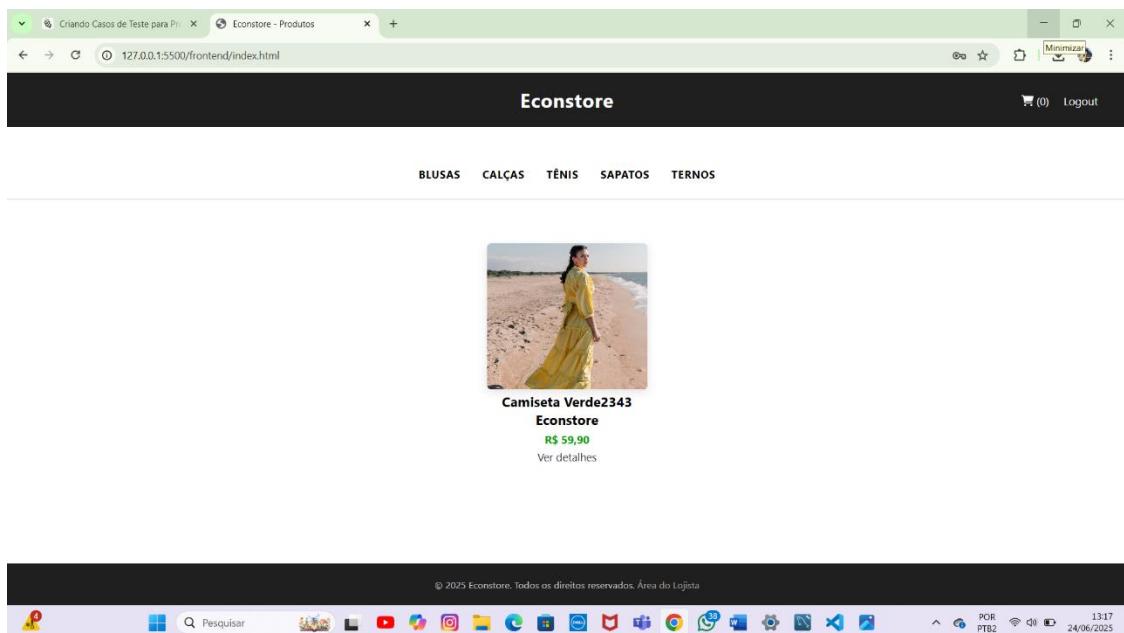
Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 -



02 -



Nome do caso de teste: CT 005 - Testar botão de logout

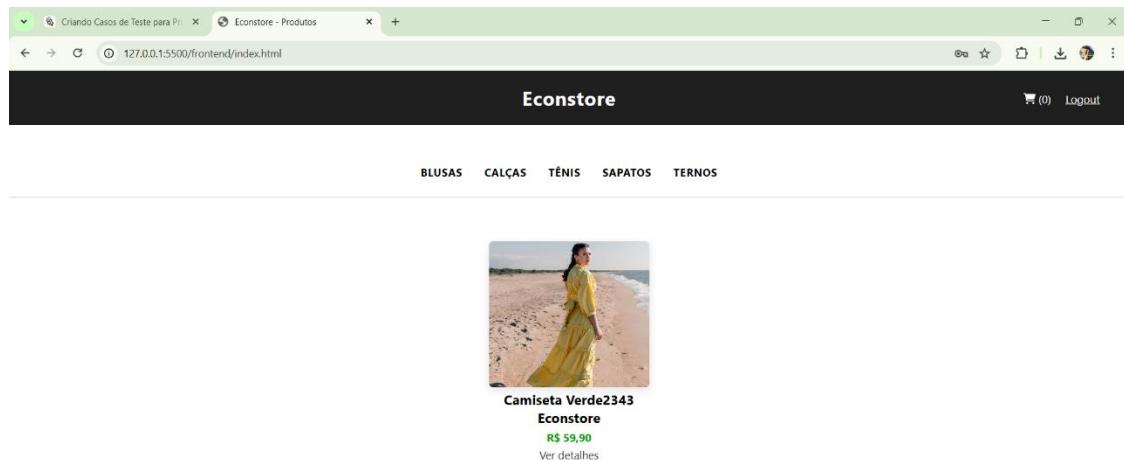
Data: 22/06/2025

Funcionalidade: Logout

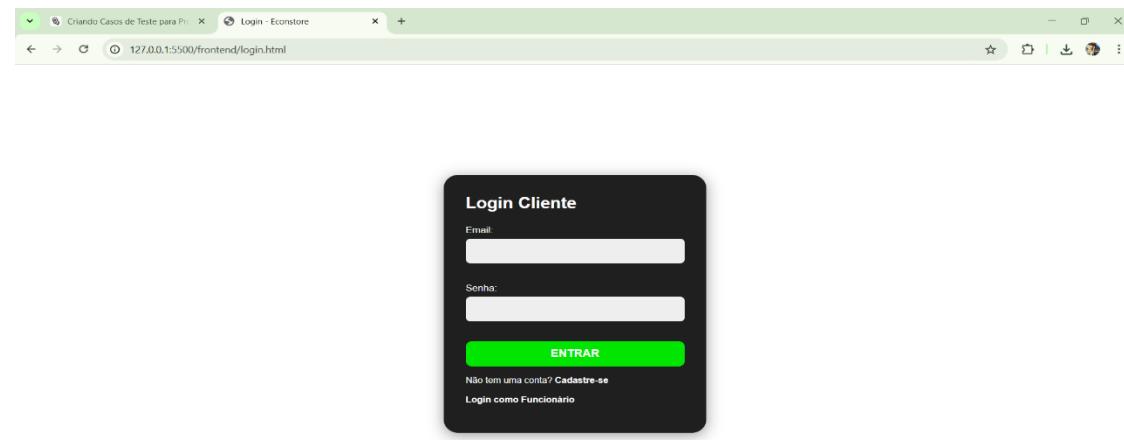
Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 -



02 -



Nome do caso de teste: CT 006 - Testar comprar um produto com quantidade maior que a disponível

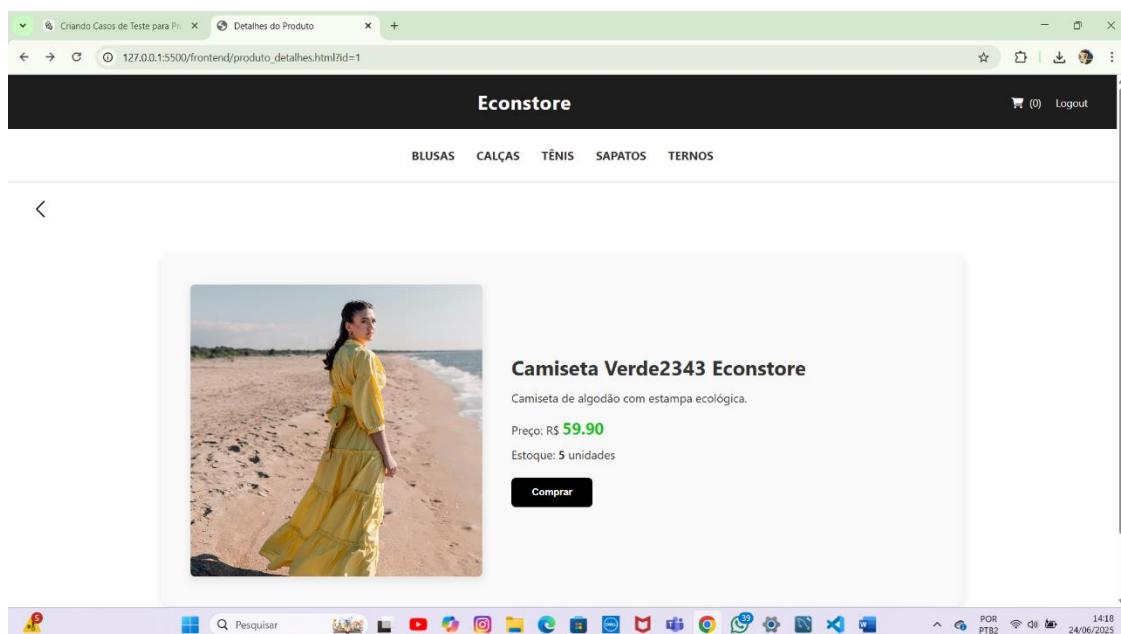
Data: 22/06/2025

Funcionalidade: Controle de estoque

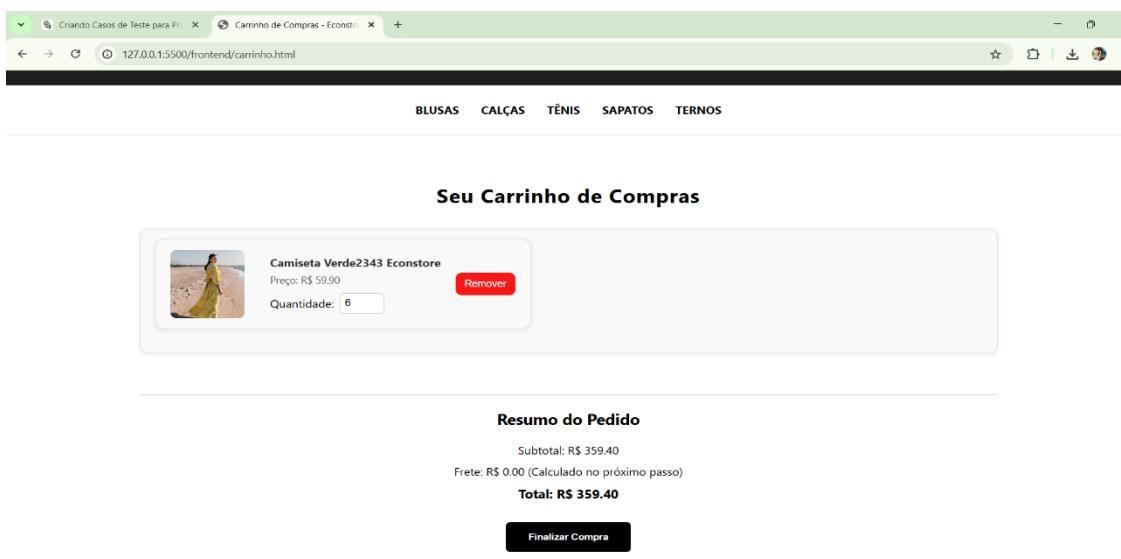
Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

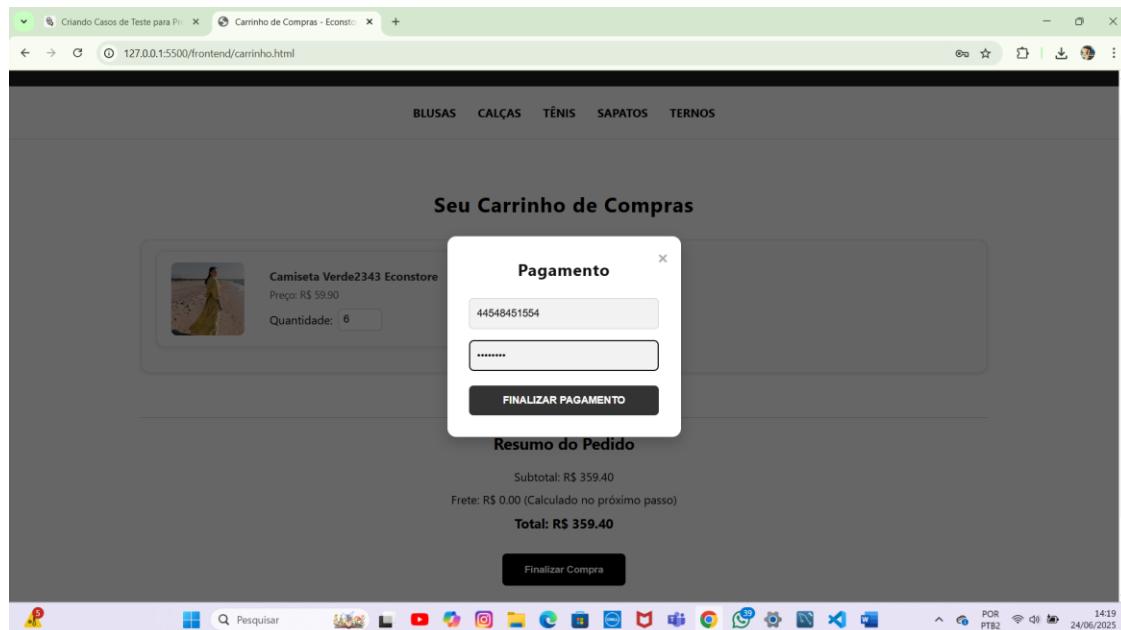
01 -



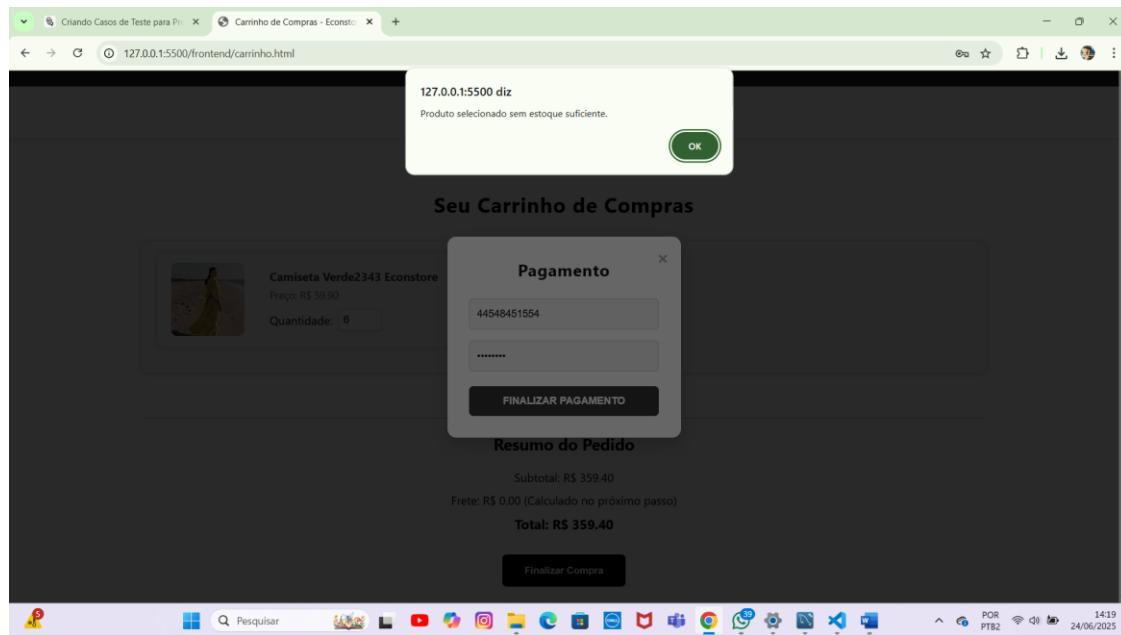
02-



03 -



04 -



Nome do caso de teste: CT 007 -- Testar remover produto do carrinho

Data: 22/06/2025

Funcionalidade: Carrinho

Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 -

The screenshot shows a web browser window with the title bar "Criando Casos de Teste para Pi: Carrinho de Compras - Econstore". The address bar shows the URL "127.0.0.1:5500/frontend/carrinho.html". The page itself is titled "Econstore" and features a navigation menu with links to "BLUSAS", "CALÇAS", "TÊNIS", "SAPATOS", and "TERNOS". A "Logout" link is also present in the top right corner. Below the menu, a section titled "Seu Carrinho de Compras" displays a single item: "Camiseta Verde2343 Econstore" with a price of "R\$ 59.90". There is a red "Remover" button and a quantity input field set to "6". Further down, a "Resumo do Pedido" section provides a breakdown of the order: "Subtotal: R\$ 359.40", "Frete: R\$ 0.00 (Calculado no próximo passo)", and "Total: R\$ 359.40". At the bottom of the screen, a Windows taskbar is visible, showing various pinned icons and the date/time "24/06/2025 14:27".

02 -

The screenshot shows a web browser window with two tabs: 'Criando Casos de Teste para Pi...' and 'Carrinho de Compras - Econstore'. The main content area displays the Econstore homepage with a navigation bar for 'BLUSAS', 'CALÇAS', 'TÊNIS', 'SAPATOS', and 'TERNOS'. Below this is a section titled 'Seu Carrinho de Compras' which contains the message 'Seu carrinho está vazio.' A horizontal line separates this from a 'Resumo do Pedido' section. In this section, it shows 'Subtotal: R\$ 0.00', 'Frete: R\$ 0.00 (Calculado no próximo passo)', and a bolded 'Total: R\$ 0.00'. A 'Finalizar Compra' button is present. The browser's taskbar at the bottom shows various pinned icons.

Nome do caso de teste: CT 008 - Testar comprar produto sem preencher os dados do cartão (pedido ficará com status de pendente)

Data: 22/06/2025

Funcionalidade: Compra

Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 -

The screenshot shows a web browser window with two tabs: 'Criando Casos de Teste para Pi...' and 'Carrinho de Compras - Econstore'. A confirmation dialog box is displayed in the center, reading '127.0.0.1:5500 diz' and 'Pedido pendente de confirmação.', with an 'OK' button. Below the dialog, the Econstore homepage is visible. It features a product card for a 'Camiseta Verde2343 Econstore' priced at 'R\$ 59.90' with 'Quantidade: 2'. To the right of the product card is a 'Pagamento' (Payment) modal with fields for 'Número Cartão' (Credit Card Number) and 'Senha Cartão' (Credit Card PIN), and a 'FINALIZAR PAGAMENTO' (Finalize Payment) button. Below the payment modal is a 'Resumo do Pedido' (Order Summary) section showing 'Subtotal: R\$ 119.80', 'Frete: R\$ 0.00 (Calculado no próximo passo)', and a bolded 'Total: R\$ 119.80'. A 'Finalizar Compra' button is located at the bottom. The browser's taskbar at the bottom shows various pinned icons.

Nome do caso de teste: CT 009 - Testar comprar produto preenchendo os dados do cartão
(pedido ficará como aprovado)

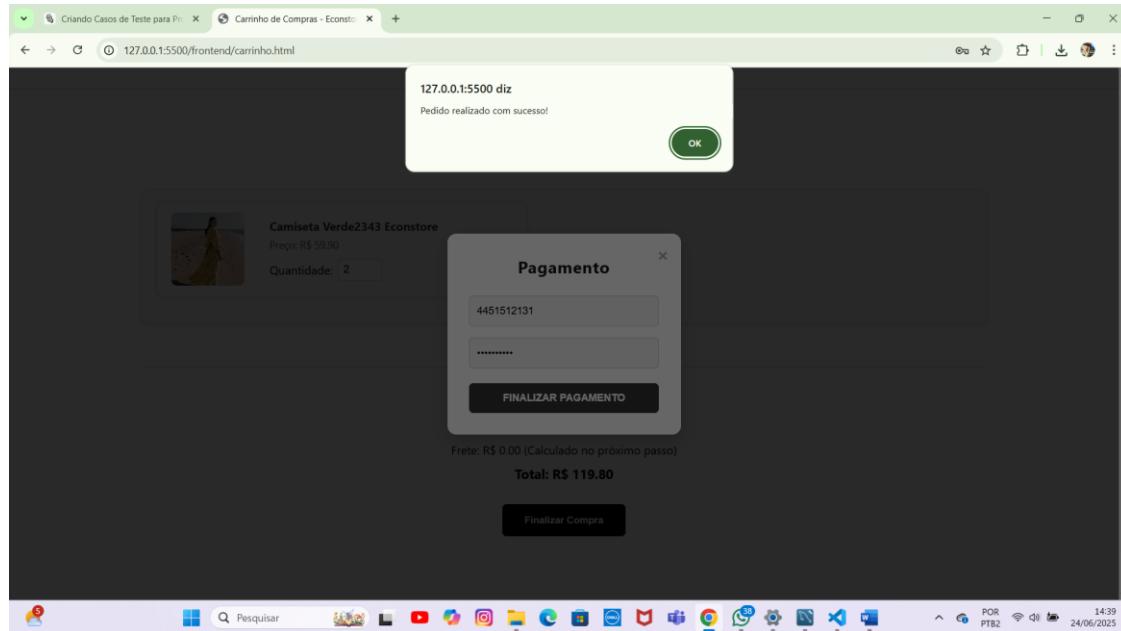
Data: 22/06/2025

Funcionalidade: Compra

Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 –



Nome do caso de teste: CT 010 - Testar ver se a quantidade de produtos disponíveis diminui após a compra

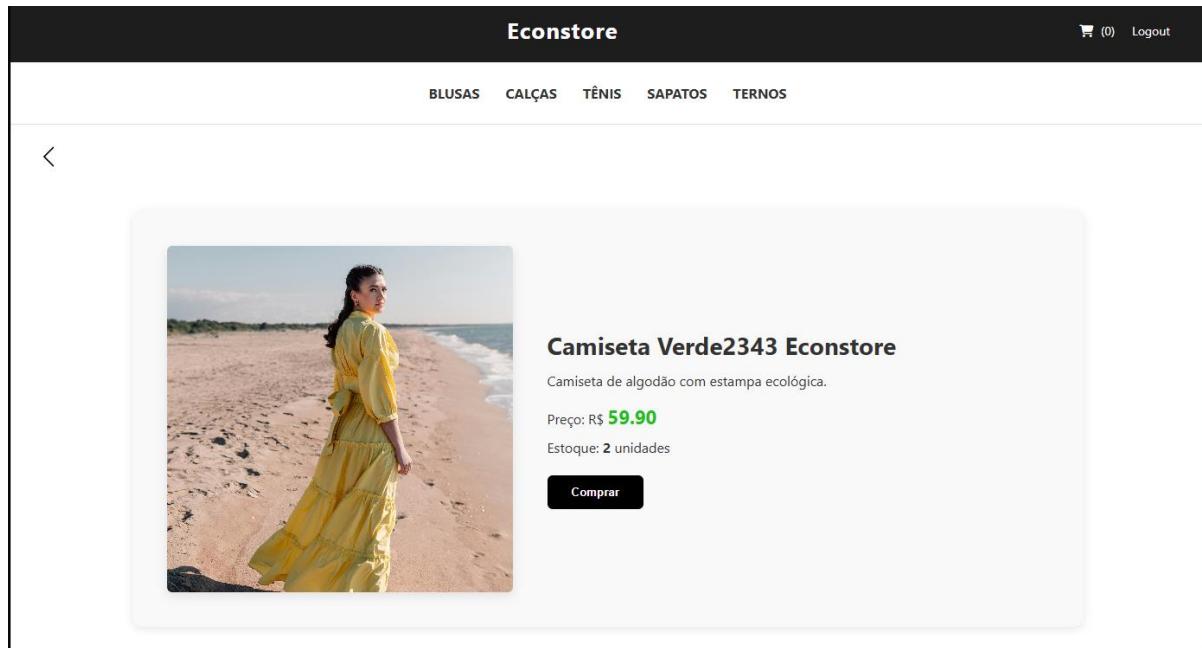
Data: 22/06/2025

Funcionalidade: Controle de Estoque

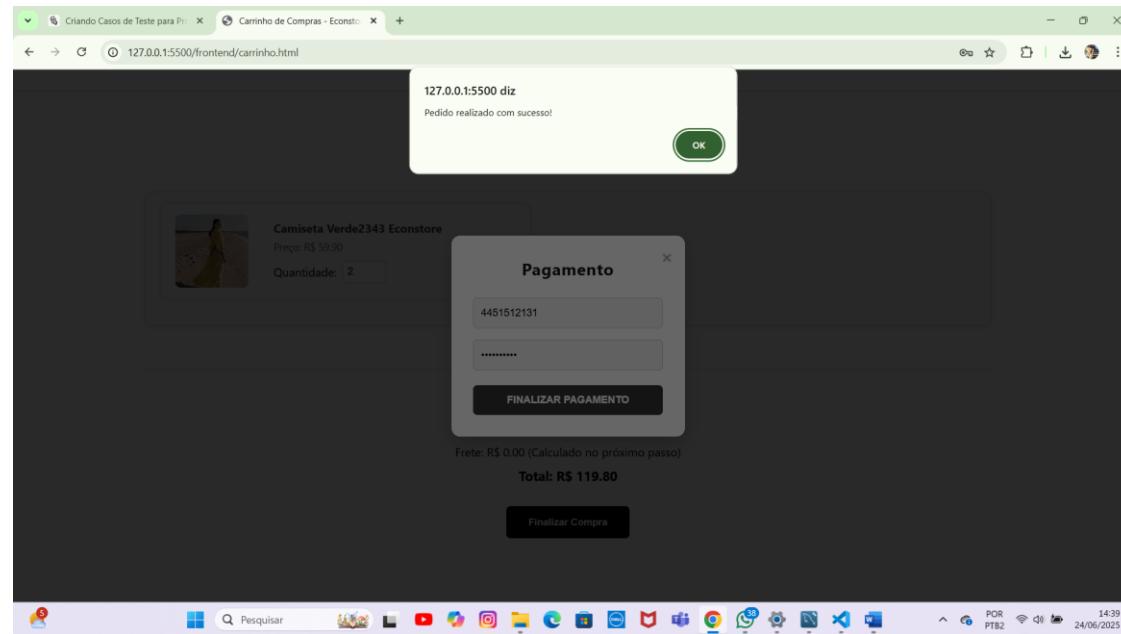
Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste

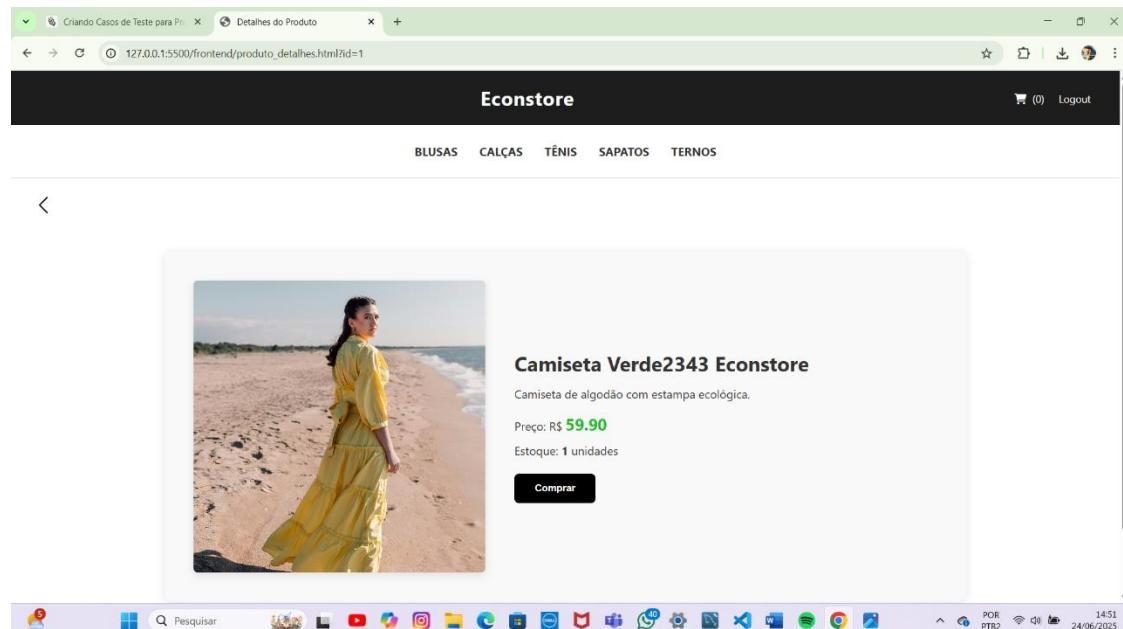
01 –



02 –



03 –



Nome do caso de teste: CT 11 - Testar login de funcionário sem preencher os campos

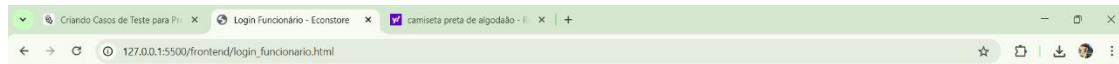
Data: 22/06/2025

Funcionalidade: Login

Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 -



02-



Nome do caso de teste: CT 012 - Testar login de funcionário com os dados inválidos

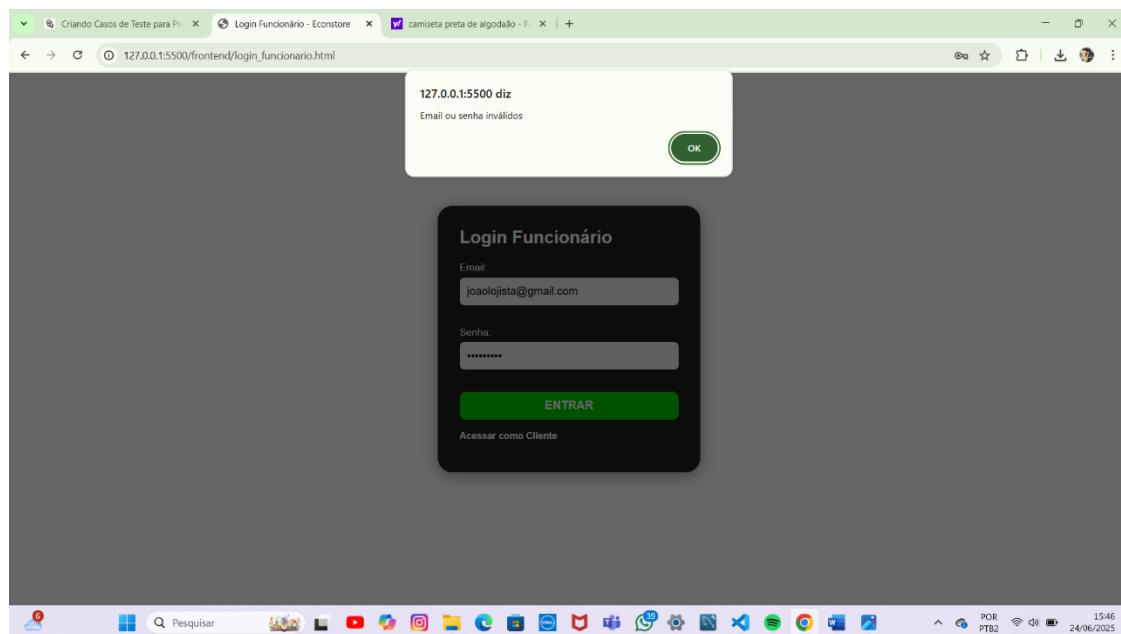
Data: 22/06/2025

Funcionalidade: Login

Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01



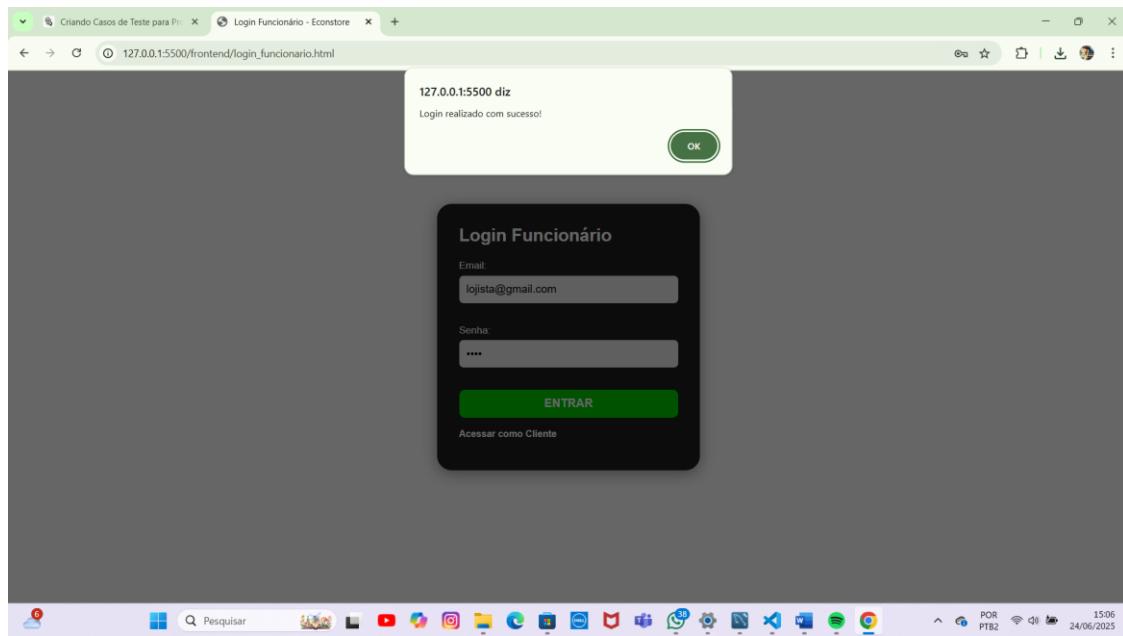
Nome do caso de teste: CT 013 - Testar login de funcionário com dados válidos Data:
22/06/2025

Funcionalidade: Login

Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 –



Nome do caso de teste: CT 014 - Testar botão de ver produtos

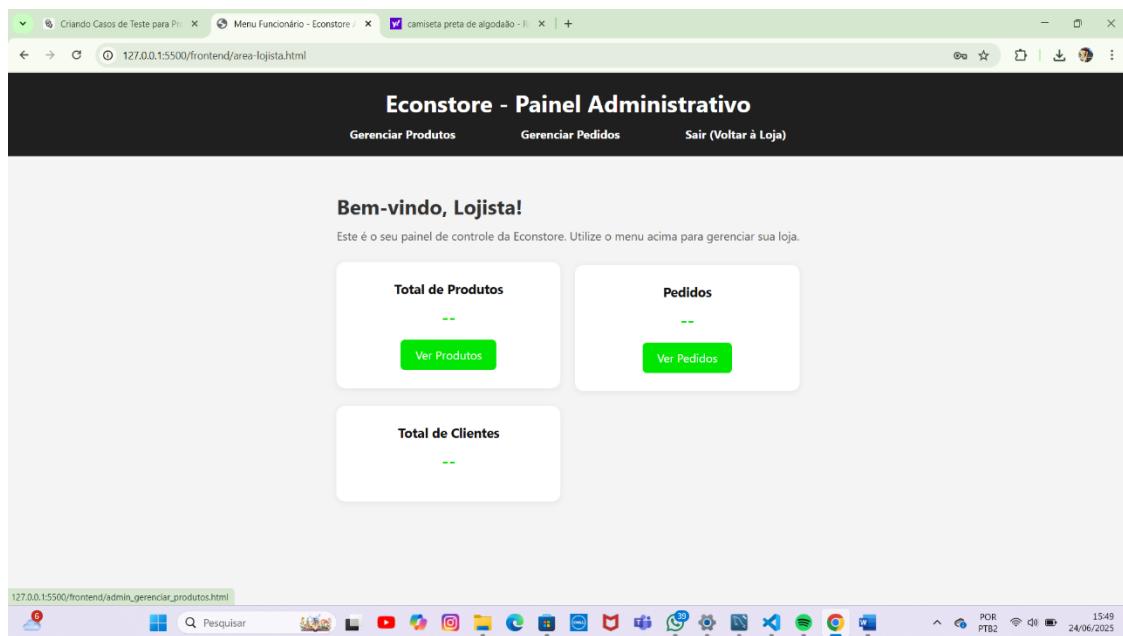
Data: 22/06/2025

Funcionalidade: Ver produtos

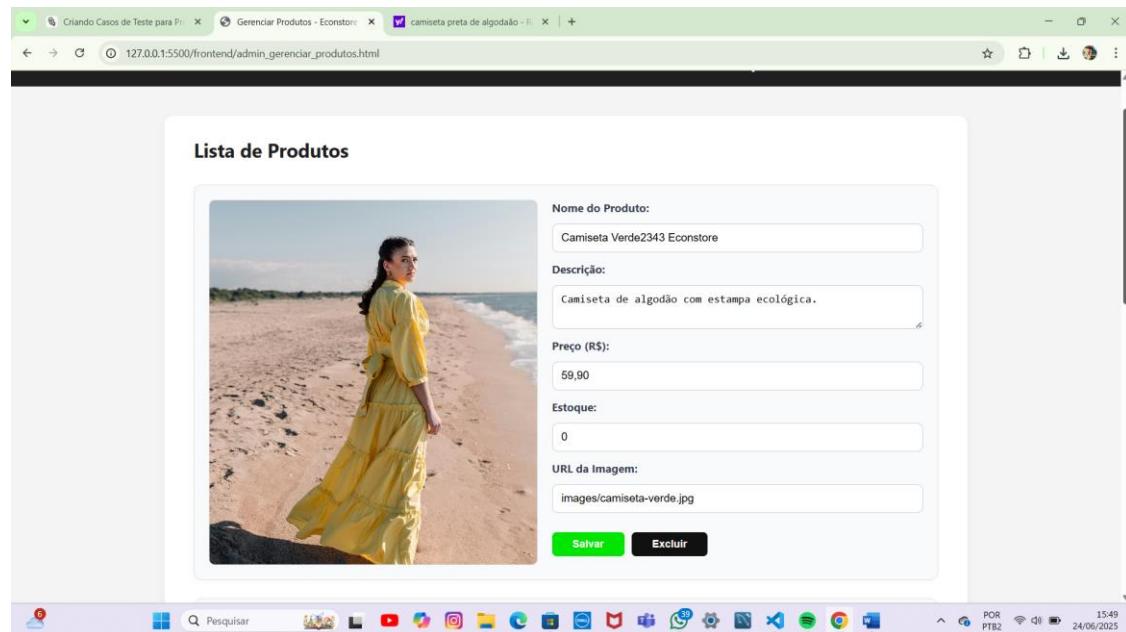
Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 -



02 –



Nome do caso de teste: CT 015 - Testar editar mudar todo um produto

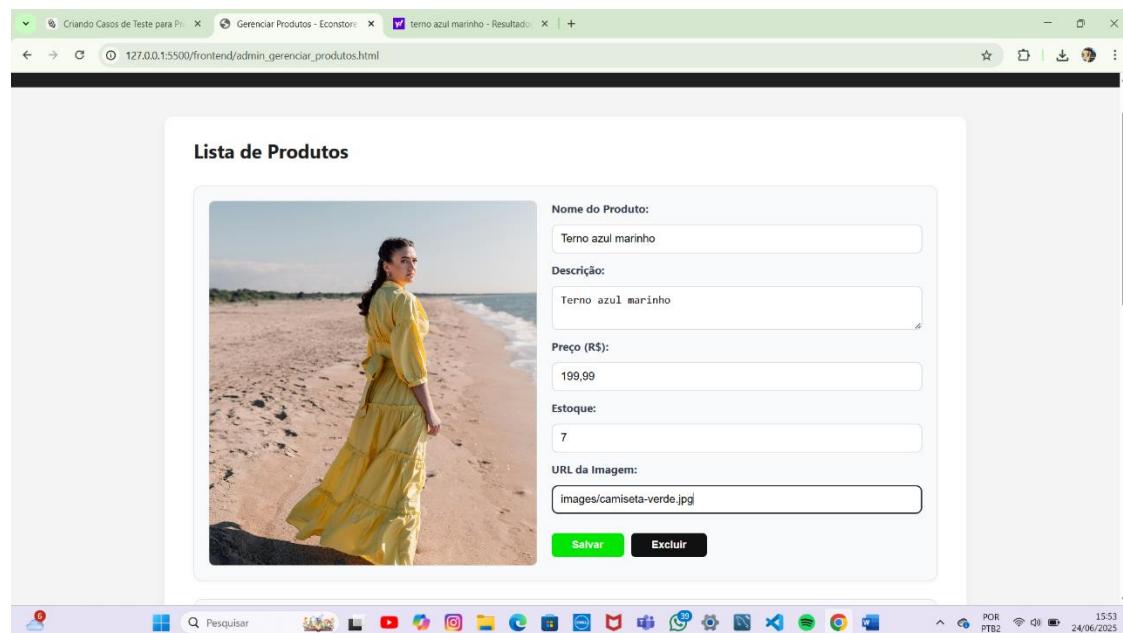
Data: 22/06/2025

Funcionalidade: Editar Produtos

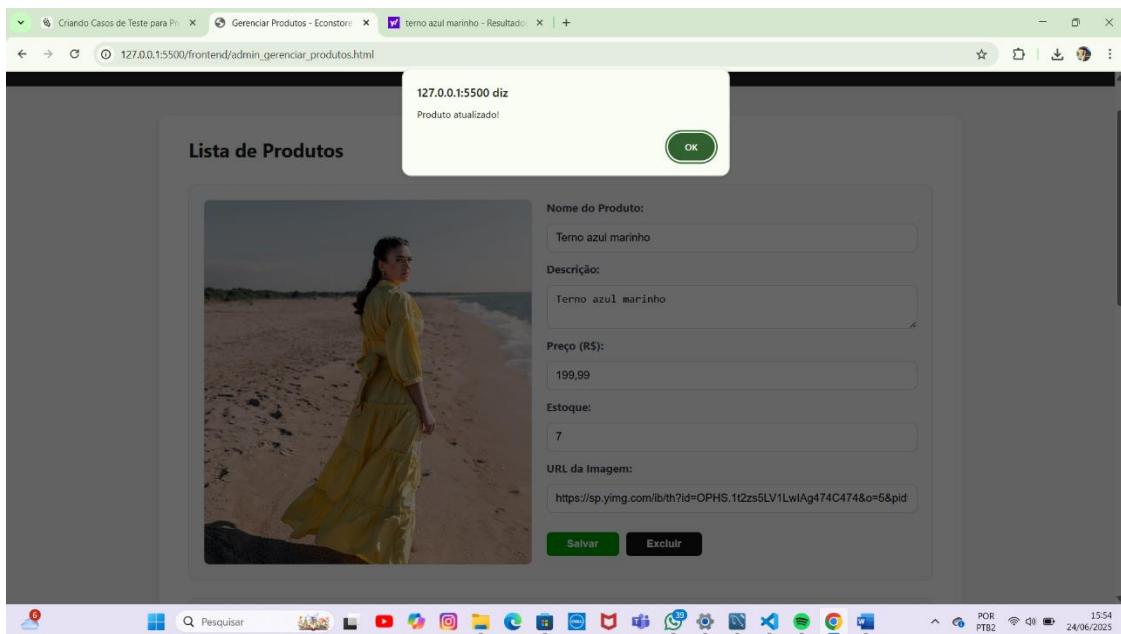
Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 -



02 -



Nome do caso de teste: CT 016 - Testar excluir um produto que tem um pedido associado a ele

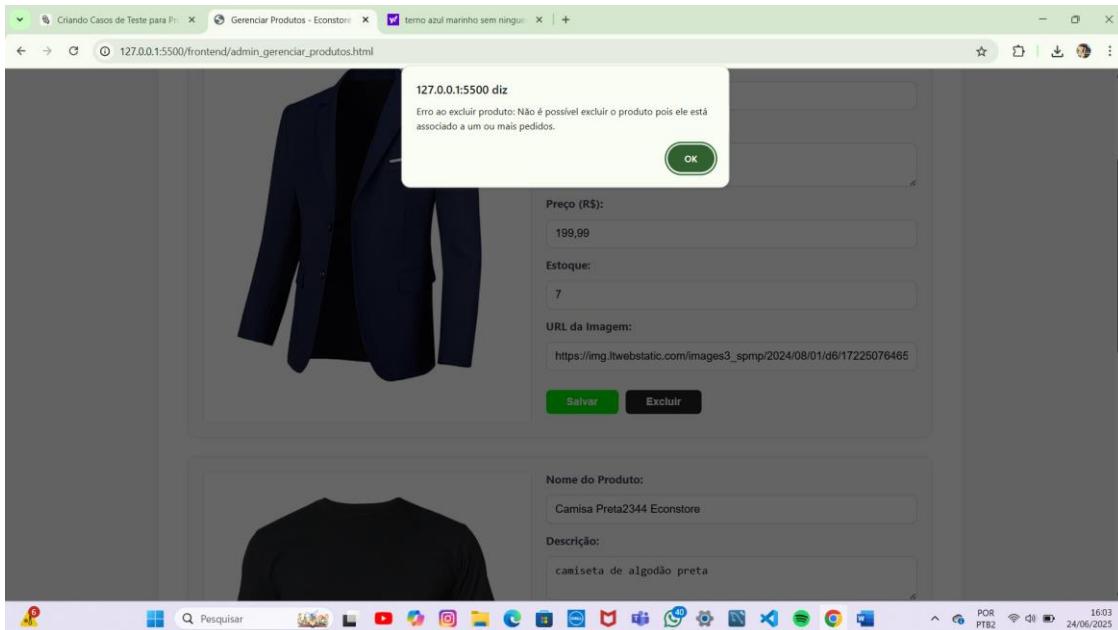
Data: 22/06/2025

Funcionalidade: Excluir Produto

Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 –



Nome do caso de teste: CT 017 - Testar excluir um produto sem pedidos

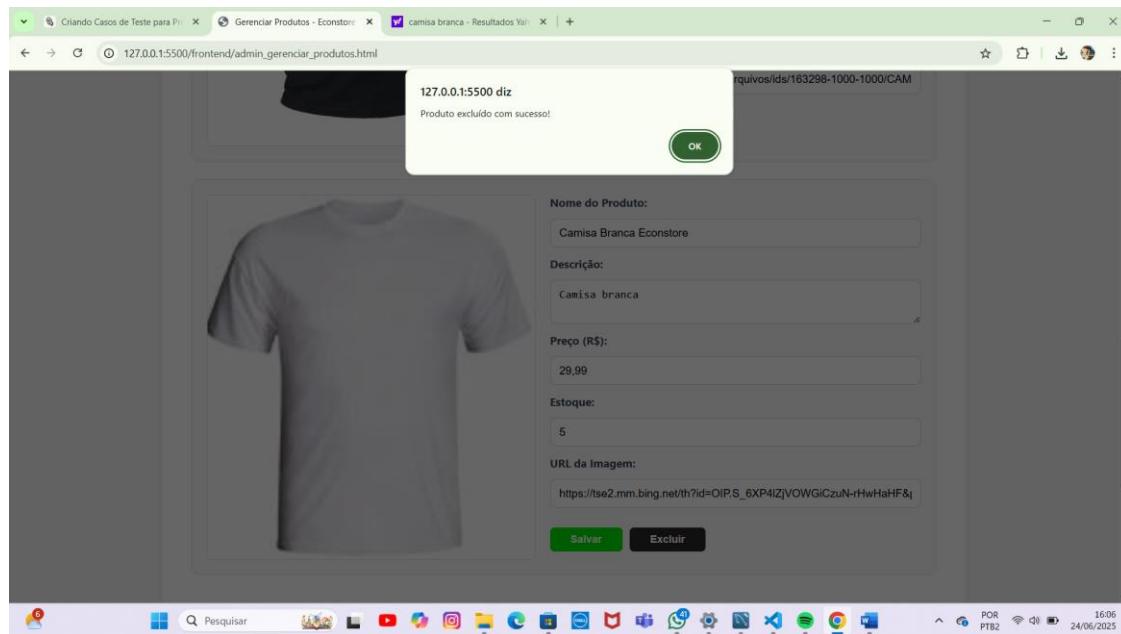
Data: 22/06/2025

Funcionalidade: Excluir Produto

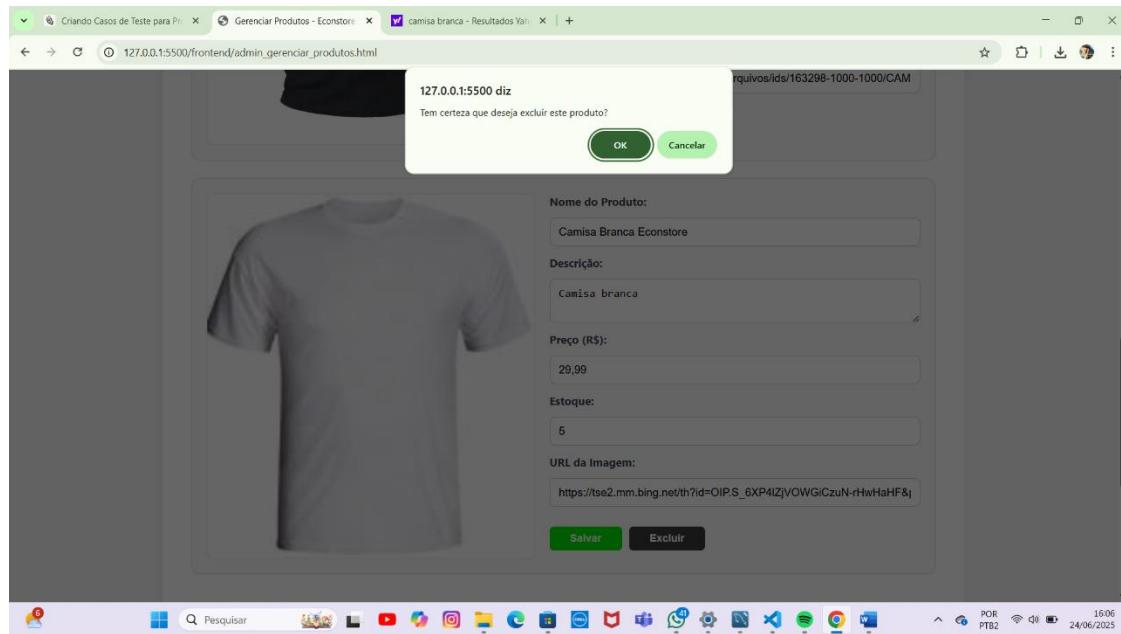
Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

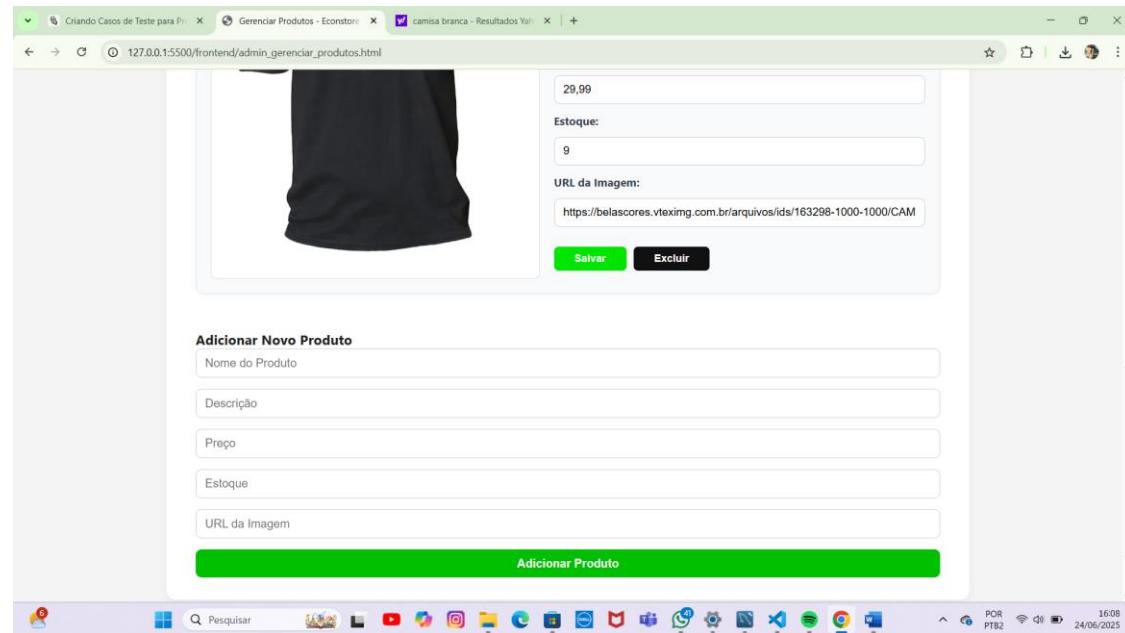
01 -



02-



03 -



Nome do caso de teste: CT 018 - Testar criar um produto com dados completos

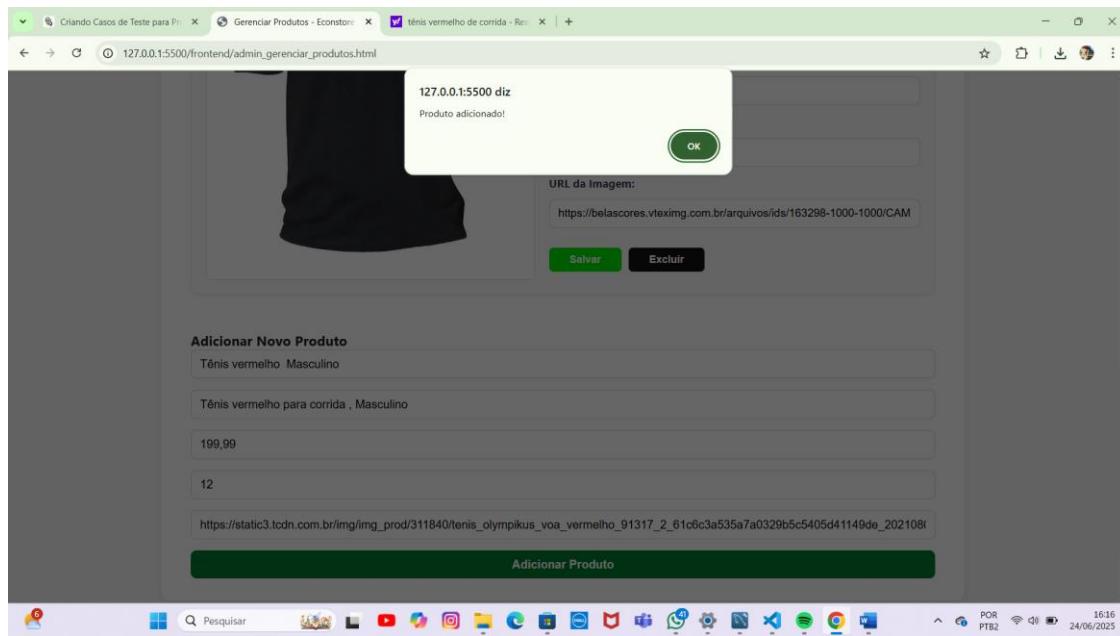
Data: 22/06/2025

Funcionalidade: Criar produto

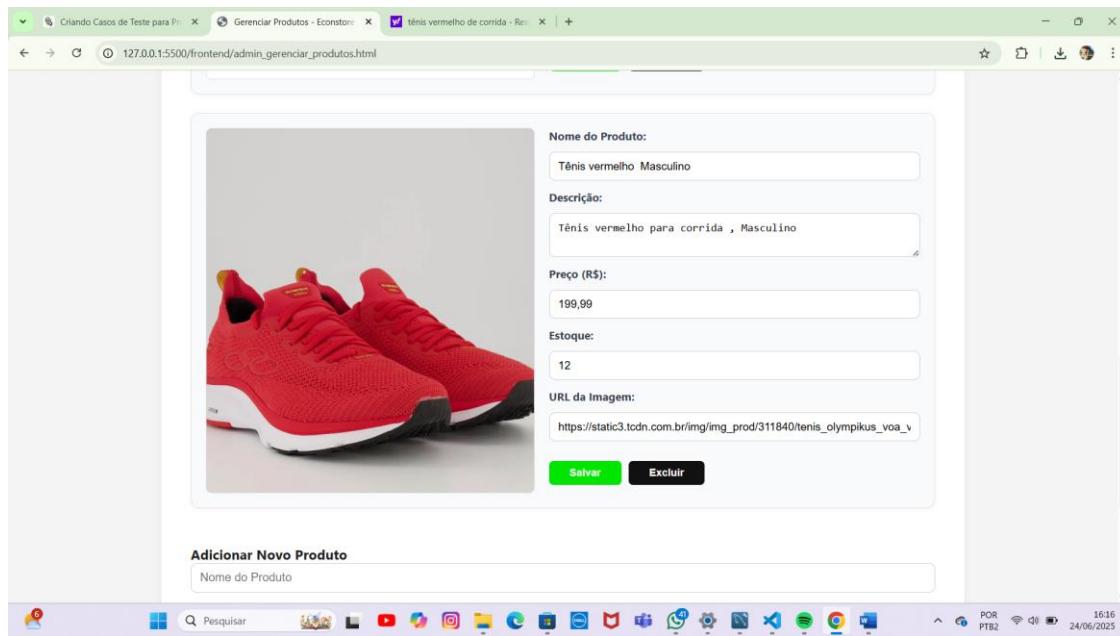
Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 -



02-



Nome do caso de teste: CT 019 - Testar comprar o produto criado

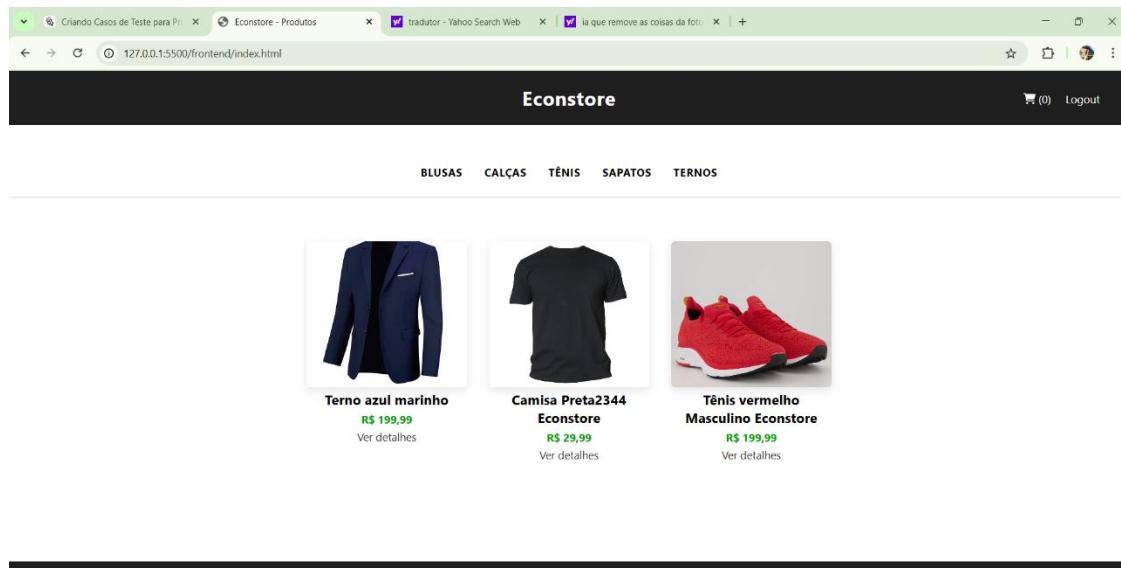
Data: 22/06/2025

Funcionalidade: Compra

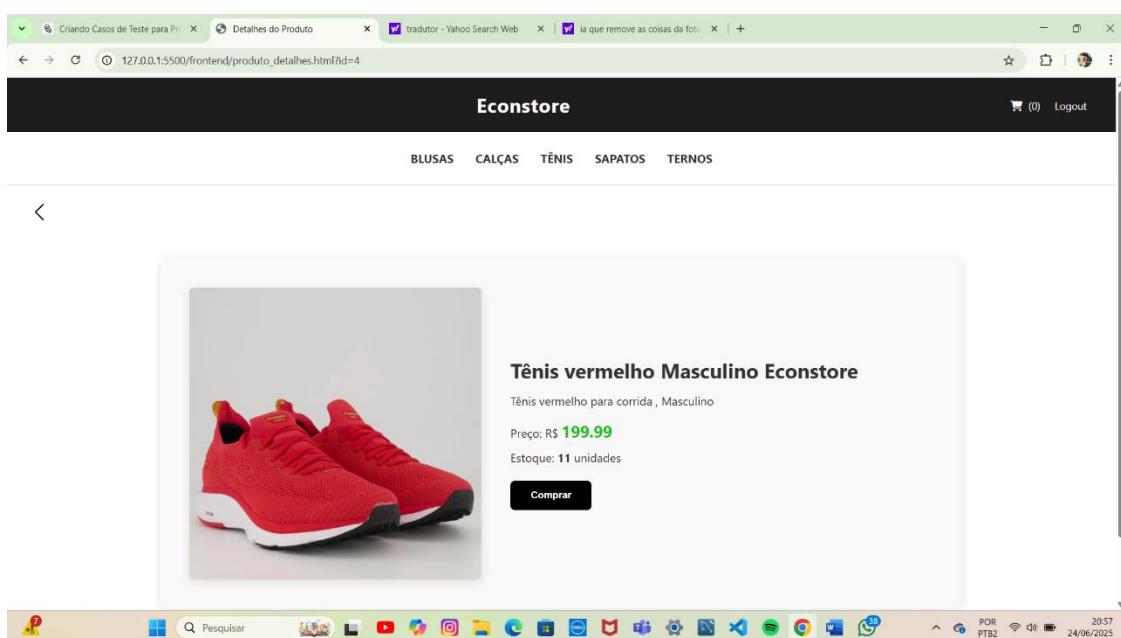
Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

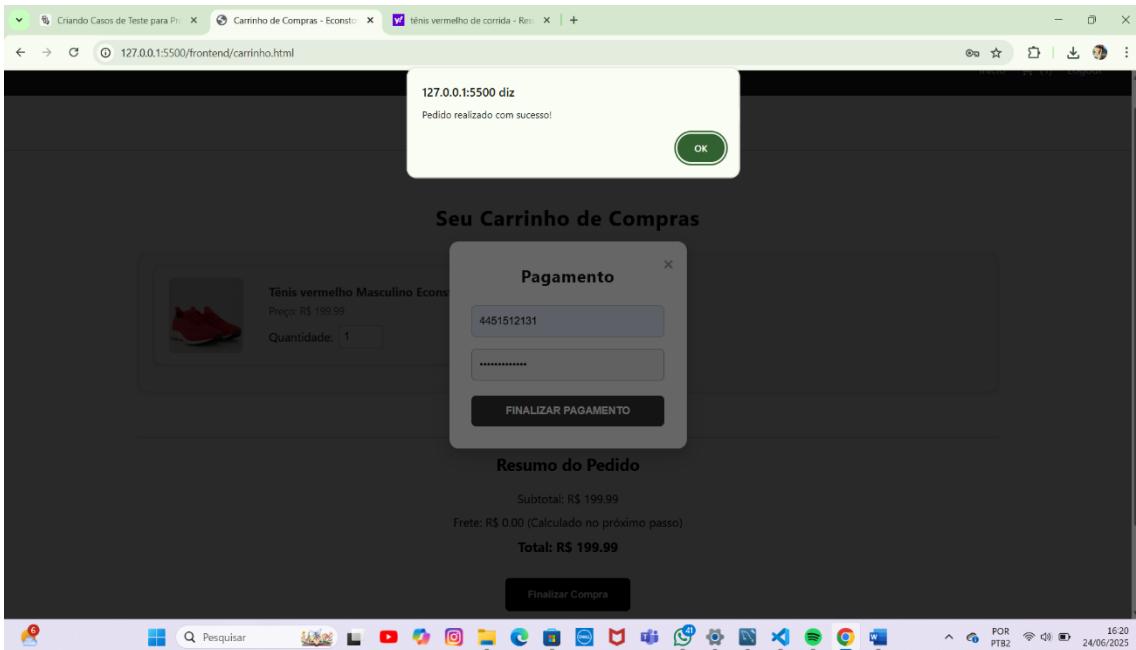
01 -



02 -



03 -



Nome do caso de teste: CT 20 - Testar botão de ver pedidos

Data: 22/06/2025

Funcionalidade: Ver pedidos

Responsável pelo teste: Gustavo Medeiros

Passos para realizar o teste:

01 -

02 -

The screenshot displays two windows of the Econstore Admin Panel.

Main Dashboard (Top Window):

- Title:** Econstore - Painel Administrativo
- Menu:** Gerenciar Produtos, Gerenciar Pedidos, Sair (Voltar à Loja)
- Bem-vindo, Lojista!**: Welcome message.
- Informações:** Total de Produtos, Pedidos, Total de Clientes.
- Ações:** Ver Produtos, Ver Pedidos.

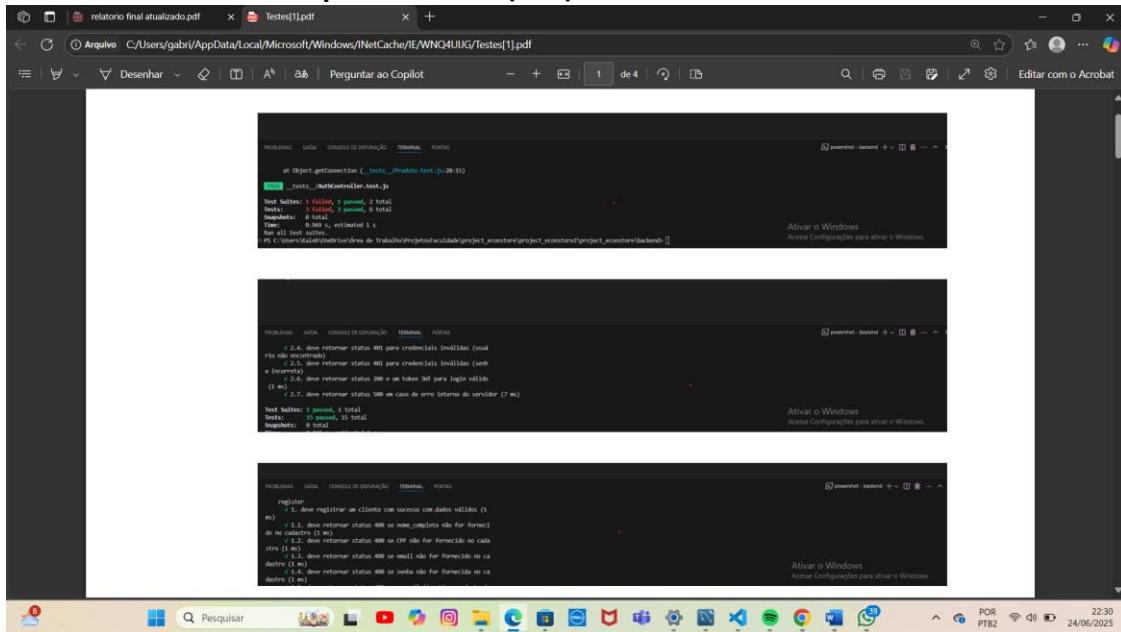
Received Orders View (Bottom Window):

- Title:** Econstore - Painel Administrativo
- Menu:** Início Admin, Gerenciar Produtos, Sair (Voltar à Loja)
- Section:** Pedidos Recebidos
- Table:** Shows a list of received orders with columns: ID, USUÁRIO, ITENS COMPRADOS, VALOR TOTAL, STATUS, and DATA.
- Data:**

ID	USUÁRIO	ITENS COMPRADOS	VALOR TOTAL	STATUS	DATA
#4	Gabriel Silva Rodrigues	Tênis vermelho Masculino Econstore (x1)	R\$ 199.99	Aprovado	24/06/2025 16:20:18
#3	Gabriel Silva Rodrigues	Camisa Preta2344 Econstore (x1), Terno azul marinho (x1)	R\$ 89.89	Aprovado	24/06/2025 15:39:14
#2	Gabriel Silva Rodrigues	Terno azul marinho (x2)	R\$ 119.80	Aprovado	24/06/2025 14:39:27
#1	Gabriel Silva Rodrigues	Terno azul marinho (x2)	R\$ 119.80	Pendente	24/06/2025 14:36:20

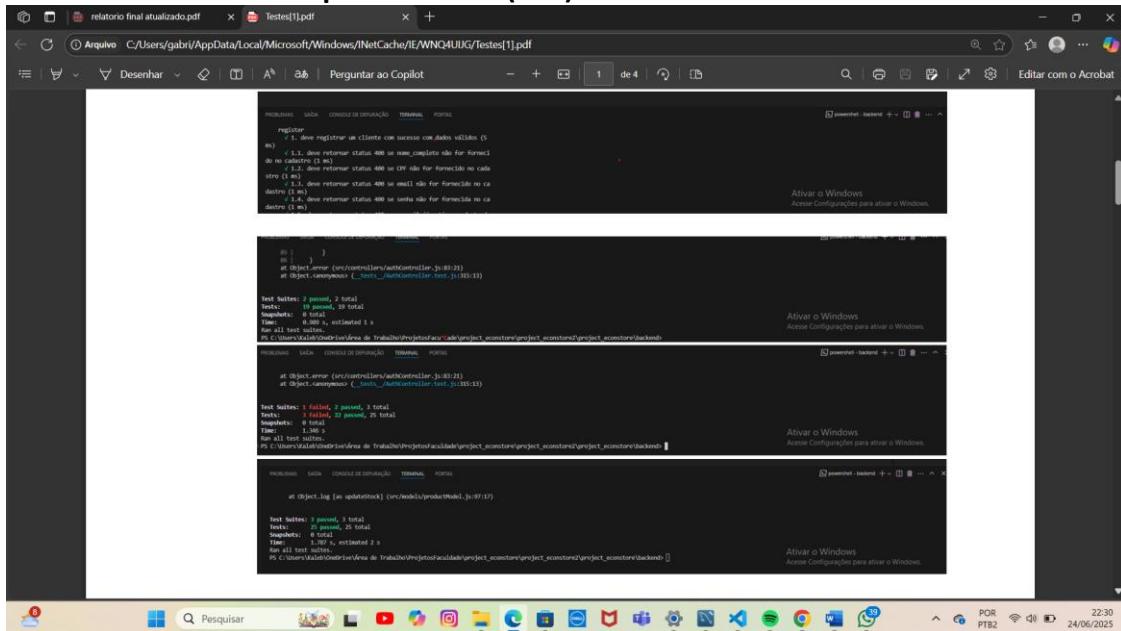
- Footer:** © 2025 Econstore Admin Panel.

Evidências de Teste - Captura de Tela (248)



Captura de Tela 248

Evidências de Teste - Captura de Tela (249)



Captura de Tela 249

Evidências de Teste - Captura de Tela (251)

The screenshot shows a Windows desktop with three terminal windows open, each displaying Jest test results. The terminals are titled 'PROBLEMAS', 'SAÍDA', 'CONSOLE DE DESENVOLVIMENTO', 'TERMINAL', and 'PORTAS'. The first terminal shows results for 'product.test.js' with 4 passed and 4 total tests. The second terminal shows results for 'FindUserById.test.js' with 4 passed and 4 total tests. The third terminal shows results for 'User.test.js' with 1 failed, 3 passed, and 4 total tests. All three terminals show a status message: 'Ativar o Windows' (Activate Windows) and 'Acesse Configurações para ativar o Windows.' (Access Settings to activate Windows.).

```
PROBLEMAS | SAÍDA | CONSOLE DE DESENVOLVIMENTO | TERMINAL | PORTAS
42 | 0 |
42 | 0 |
at object.error [as getAllProducts] (src/models/product.js:48:21)
at Object.error [as findById] (src/models/userMethod.js:57:21)
at Object.error [as findById] (src/models/test.js:208:13)

Test Suites: 1 passed, 4 total
Tests: 4 passed, 4 total
Snapshots: 0 total
Time: 1.00 s
Ran all test suites.
PS C:\Users\Gabri\OneDrive\Área de Trabalho\ProjetosFaculdade\project_econstore\project_econstore2\project_econstore\backend\b

PROBLEMAS | SAÍDA | CONSOLE DE DESENVOLVIMENTO | TERMINAL | PORTAS
at object.error [as findById] (src/models/userMethod.js:57:21)
at Object.error [as findById] (src\models\userMethod.test.js:208:13)

Test Suites: 1 passed, 4 total
Tests: 4 passed, 4 total
Snapshots: 0 total
Time: 1.00 s
Ran all test suites.
PS C:\Users\Gabri\OneDrive\Área de Trabalho\ProjetosFaculdade\project_econstore\project_econstore2\project_econstore\backend\b

PROBLEMAS | SAÍDA | CONSOLE DE DESENVOLVIMENTO | TERMINAL | PORTAS
at object.error [as findById] (src\models\userMethod.js:57:21)
at Object.error [as findById] (src\models\userMethod.test.js:208:13)

Test Suites: 1 failed, 3 passed, 4 total
Tests: 4 failed, 3 passed, 4 total
Snapshots: 0 total
Time: 1.00 s
Ran all test suites.
PS C:\Users\Kaleb\OneDrive\Área de Trabalho\ProjetosFaculdade\project_econstore\project_econstore2\project_econstore\backend\b
```

CapturadeTela251

Evidências de Teste - Captura de Tela (252)

The screenshot shows a Windows desktop with three terminal windows open, each displaying Jest test results. The terminals are titled 'PROBLEMAS', 'SAÍDA', 'CONSOLE DE DESENVOLVIMENTO', 'TERMINAL', and 'PORTAS'. The first terminal shows results for 'ProductController.test.js' with 4 passed and 4 total tests. The second terminal shows results for 'User.test.js' with 4 passed and 4 total tests. The third terminal shows results for 'User.test.js' with 1 failed, 4 passed, and 5 total tests. All three terminals show a status message: 'Ativar o Windows' (Activate Windows) and 'Acesse Configurações para ativar o Windows.' (Access Settings to activate Windows.).

```
PROBLEMAS | SAÍDA | CONSOLE DE DESENVOLVIMENTO | TERMINAL | PORTAS
console.error
    Erro ao atualizar produto: Error: Falha de DB durante update
    at Object.<anonymous> (C:\Users\Kaleb\OneDrive\Área de Trabalho\ProjetosFaculdade\project_econstore\project_econstore2\project_econstore\backend\_tests\_ProductController.test.js:340:31)
        at Promise.finally.completed (C:\Users\Kaleb\OneDrive\Área de Trabalho\ProjetosFaculdade\project_econstore\project_econstore2\project_econstore\backend\node_modules\jest-circus\build\jestAdapterInit.js:1559:28)
            at new Promise (<anonymous>)
            at callAsyncCircusFn (C:\Users\Kaleb\OneDrive\Área de Trabalho\ProjetosFaculdade\project_econstore\project_econstore2\project_econstore\backend\b
at object.error [as findById] (src\models\userMethod.js:57:21)
at Object.error [as findById] (src\models\userMethod.test.js:208:13)

Test Suites: 4 passed, 4 total
Tests: 4 passed, 4 total
Snapshots: 0 total
Time: 1.00 s
Ran all test suites.
PS C:\Users\Kaleb\OneDrive\Área de Trabalho\ProjetosFaculdade\project_econstore\project_econstore2\project_econstore\backend\b

at object.error [as findById] (src\models\userMethod.js:57:21)
at Object.error [as findById] (src\models\userMethod.test.js:208:13)

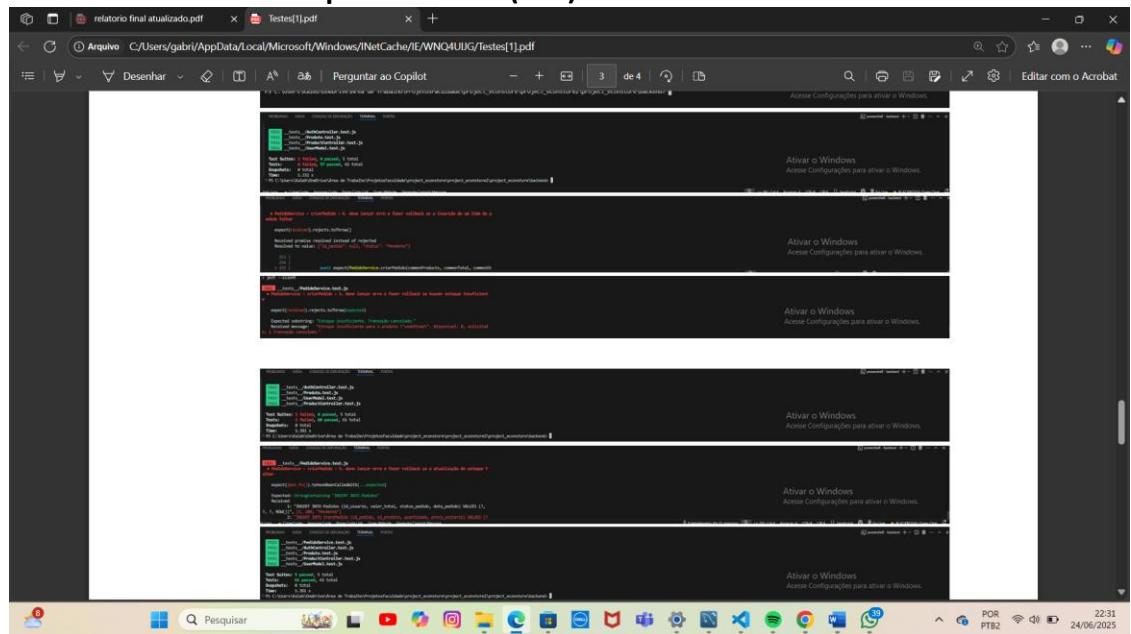
Test Suites: 4 passed, 4 total
Tests: 4 passed, 4 total
Snapshots: 0 total
Time: 1.00 s
Ran all test suites.
PS C:\Users\Kaleb\OneDrive\Área de Trabalho\ProjetosFaculdade\project_econstore\project_econstore2\project_econstore\backend\b

at object.error [as findById] (src\models\userMethod.js:57:21)
at Object.error [as findById] (src\models\userMethod.test.js:208:13)

Test Suites: 1 failed, 4 passed, 5 total
Tests: 5 failed, 4 passed, 5 total
Snapshots: 0 total
Time: 1.00 s
Ran all test suites.
PS C:\Users\Kaleb\OneDrive\Área de Trabalho\ProjetosFaculdade\project_econstore\project_econstore2\project_econstore\backend\b
```

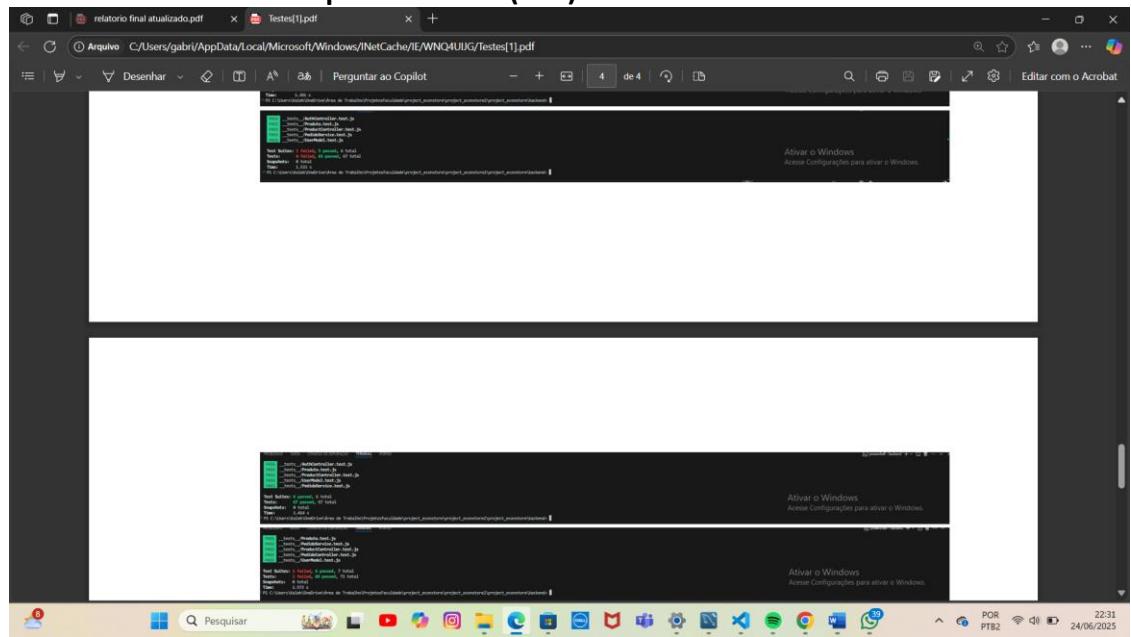
CapturadeTela252

Evidências de Teste - Captura de Tela (253)



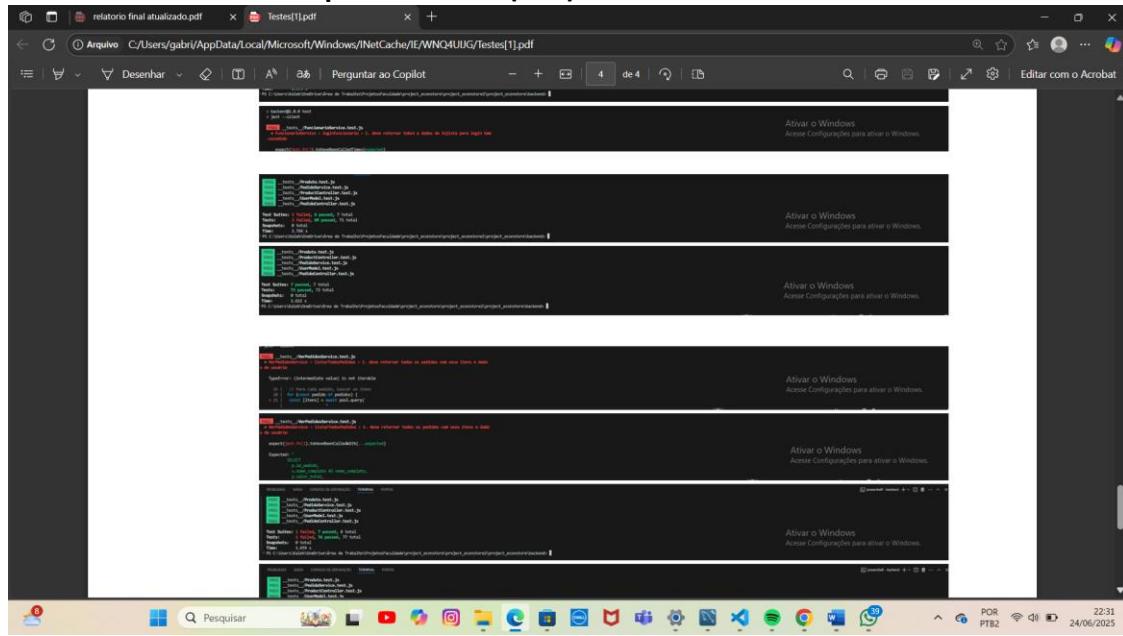
CapturadeTela253

Evidências de Teste - Captura de Tela (254)



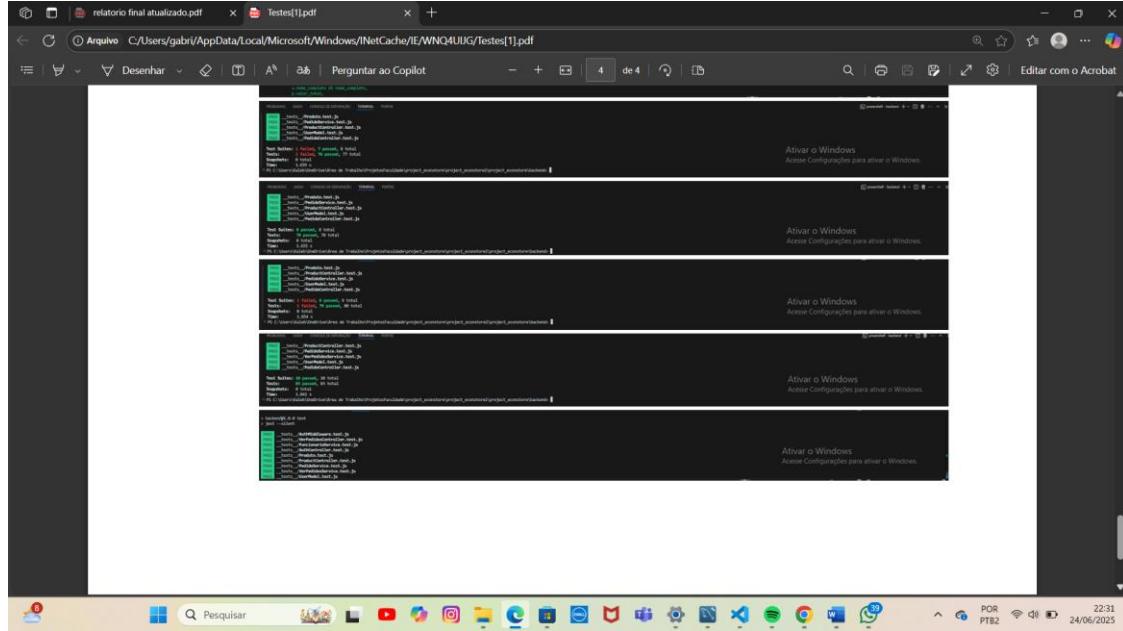
CapturadeTela254

Evidências de Teste - Captura de Tela (255)



CapturadeTela255

Evidências de Teste - Captura de Tela (256)



CapturadeTela256

POSTMAN:

GET API:

The screenshot shows the Postman application interface. At the top, there is a header bar with a 'HTTP' icon, the URL 'http://localhost:3001/api', a 'Save' button, and a 'Share' button. Below the header, a search bar contains 'GET' and the URL 'http://localhost:3001/api'. To the right of the search bar is a 'Send' button. Underneath the search bar, there are tabs for 'Params', 'Authorization', 'Headers (6)', 'Body', 'Scripts', 'Settings', and 'Cookies'. The 'Headers (6)' tab is currently selected. Below these tabs is a section titled 'Query Params' with a table:

Key	Value	Description	Bulk Edit
Key	Value	Description	

At the bottom of the interface, there are tabs for 'Body', 'Cookies', 'Headers (8)', and 'Test Results'. The 'Test Results' tab is currently selected. It displays a green '200 OK' status bar with performance metrics: 7 ms, 311 B, and a globe icon. Below the status bar, there are buttons for 'JSON', 'Preview', 'Visualize', and other tools. A JSON response is shown in the preview area:

```
1 {  
2 |   "message": "Bem-vindo à API da Econstore!"  
3 }
```

GET API AUTH:

The screenshot shows the Postman interface with a failed API call. The URL is `http://localhost:3001/api/auth`. The response status is **404 Not Found**, with a duration of 6 ms and a body size of 451 B. The response content is an HTML error page with the message "Cannot GET /api/auth".

GET API PEDIDOS:

The screenshot shows the Postman interface with a failed API call. The URL is `http://localhost:3001/api/pedidos`. The response status is **404 Not Found**, with a duration of 4 ms and a body size of 454 B. The response content is an HTML error page with the message "Cannot GET /api/pedidos".

GET API VER PEDIDOS:

The screenshot shows the Postman interface with the following details:

- Method:** GET
- URL:** http://localhost:3001/api/ver-pedidos
- Headers:** (6) (highlighted)
- Body:** This request does not have a body
- Params:** none
- Authorization:** None
- Scripts:** None
- Settings:** None
- Cookies:** None

Below the request details, the response is shown:

- Status:** 200 OK
- Time:** 11 ms
- Size:** 473 B
- Content:** (HTML view selected)

```
1  [{"id_pedido":1,"nome_completo":"wadwad","valor_total":"59.90","status_pedido":"Pendente",
  "data_pedido":"2025-06-25T13:25:14.000Z","itens":[{"quantidade":1,"nome_produto":"Camiseta
2  Verde2343 Econstore"}]}]
```

GET API PRODUCTS:

The screenshot shows a Postman interface for a GET request to `http://localhost:3001/api/products`. The 'Body' tab is selected, showing the message 'This request does not have a body'. Below the request details, the response section is visible, showing a 200 OK status with a response time of 6 ms and a size of 602 B. The response body is displayed in HTML, showing a JSON array of product objects. One object is highlighted with line numbers 1 and 2.

```
1 [ {"id_produto":1,"nome_produto":"Camiseta Verde2343 Econstore","descricao":"Camiseta de algodão com estampa ecológica.", "preco":59.9,"quantidade_estoque":4,"id_categoria":null,"imagem_url":"images/camiseta-verde.jpg","data_criacao":"2025-06-25T13:20:27.000Z","data_atualizacao":"2025-06-25T13:25:14.000Z","nome_categoria":null}]
```

5. Conclusão e Lições Aprendidas

A execução dos testes no sistema Econstore permitiu avaliar de forma abrangente a qualidade do software desenvolvido, abrangendo funcionalidades essenciais tanto para o cliente quanto para o funcionário (lojista). De modo geral, o sistema demonstrou bom desempenho nos principais fluxos de uso, especialmente no processo de login, cadastro, gerenciamento de produtos e finalização de pedidos.

A estrutura de testes unitários com Jest foi implementada e validada com sucesso no projeto Econstore. Os testes foram executados com 100% de aprovação, cobrindo funcionalidades críticas do sistema.

Ao longo do processo de testes, o grupo adquiriu diversas lições importantes:

- **Importância de testar desde as etapas iniciais:** Iniciar os testes logo nas primeiras versões evitou o acúmulo de erros em funcionalidades mais complexas.

- **Relevância de um bom planejamento:** Ter definido previamente os casos de teste ajudou a manter o foco e a organização.
- **Valor da documentação de testes:** Ter registros claros e estruturados sobre cada teste executado facilitou a identificação de padrões de erro e ajudou na priorização de correções.

6. Link para o repositório no GitHub

Link para o Repositório no GitHub: <https://github.com/gugutoads/teste-finalll.git>