

# QUALIDADE DE SOFTWARE

Letícia de Oliveira Cardoso

Análise de Qualidade

Hortolândia – SP  
2022

# RESUMO

O desenvolvimento de um sistema é construído por diversos processos, neste trabalho será apresentado um dos processos essenciais que é o teste de software. Estes testes verificam se todas as funcionalidades implementadas estão funcionando corretamente e quais as falhas que o sistema apresenta antes do produto chegar ao usuário final. O projeto utiliza a EBAC Shop para fazer a análise de qualidade a partir das histórias de usuário: adicionar item ao carrinho, login na plataforma e API de cupons. Serão elaborados estratégias e casos de testes, e será apresentado testes automatizados e de performance.

# SUMÁRIO

1. [RESUMO 2](#_bookmark0)
2. [SUMÁRIO 3](#_bookmark1)
3. [INTRODUÇÃO 4](#_bookmark2)
4. [O PROJETO 5](#_bookmark3)
   1. [Estratégia de teste 5](#_bookmark4)
   2. [Critérios de aceitação 5](#_bookmark5)
      1. [História de usuário 1: [US-0001] – Adicionar item ao carrinho 5](#_bookmark6)
      2. [História de usuário 2: [US-0002] – Login na plataforma 6](#_bookmark7)
      3. [História de usuário 3: [US-0003] – API de cupons 7](#_bookmark8)
   3. [Casos de testes 8](#_bookmark9)
      1. [História de usuário 1: 8](#_bookmark10)
      2. [História de usuário 2: 9](#_bookmark11)

[4.3.1 História de usuário 3: API de Cupom 9](#_bookmark12)

* 1. [Repositório no Github 10](#_bookmark13)
  2. [Testes automatizados 10](#_bookmark14)
  3. [Integração contínua 11](#_bookmark15)
  4. [Testes de performance 11](#_bookmark16)

1. [CONCLUSÃO 12](#_bookmark17)
2. [REFERÊNCIAS BIBLIOGRÁFICAS 12](#_bookmark18)

# INTRODUÇÃO

Este trabalho traz a análise de qualidade do sistema EBAC Shop, onde foi estudado 3 principais funcionalidades do sistema: a primeira é adicionar produtos no carrinho, onde o usuário adicionará todos os itens que deseja para depois finalizar a compra dos itens e onde haverá algumas restrições em cada compra e cupons de desconto; a segunda funcionalidade é a de login na plataforma, que tem validação de mensagem e um limite de tentativas para conseguir realizar o seu login; e por fim, a funcionalidade de API de cupons, que permite cadastro e listagem dos cupons. Para isso, serão elaboradas estratégias de testes, critérios de aceitação com Gherkin, casos de testes que serão disponibilizados no GitHub e testes automatizados de UI, API, integração contínua e de performance

# O PROJETO

# Estratégia de teste

Diagrama

Descrição gerada automaticamente

# Critérios de aceitação

# História de usuário 1: [US-0001] – Adicionar item ao carrinho

Critérios de aceitação:

* Não é permitido inserir mais de 10 itens de um mesmo produto ao carrinho;
* Os valores não podem ultrapassar a R$ 990,00;
* Valores entre R$ 200 e R$ 600 ganham cupom de 10%;
* Valores acima de R$ 600 ganham cupom de 15%.

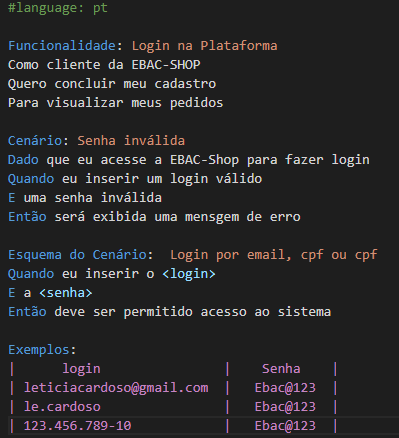
Texto

Descrição gerada automaticamente

# História de usuário 2: [US-0002] – Login na plataforma

Critérios de aceitação:

* Somente usuários ativos podem fazer login;
* Deve exibir uma mensagem de erro caso o usuário erre o login e senha;
* Login deve permitir e-mail, nome de usuário ou cpf;
* Se o usuário errar por 3 vezes a senha, deve travar por 15 minutos o login.



# História de usuário 2: [US-0003] – API de cupons

Critérios de aceitação:

* Deve listar todos os cupons cadastrado ou listar buscando por ID do cupom;
* Deve cadastrar os cupons com os campos obrigatórios abaixo:
  + Código do cupom: Exemplo: “Ganhe10”

- Valor: “10.00”

* + Tipo do desconto: “fixed\_product”

-Descrição: “Cupom de teste”

* Nome do cupom não pode ser repetido;
* Os outros campos são opcionais.

Texto

Descrição gerada automaticamente

# Casos de testes

# História de usuário 1:

CT01: Inserir 9 unidades do mesmo produto no carrinho, deve permitir a realização da compra;

CT02: Inserir 10 unidades do mesmo produto no carrinho, deve permitir a realização da compra;

CT03: Inserir 11 unidades do mesmo produto no carrinho, não deve permitir a realização da compra e exibir mensagem de erro;

CT04: Inserir produtos até o valor R$989,99, deve permitir a finalização da compra;

CT05: Inserir produtos até o valor R$990,00, deve permitir a finalização da compra;

CT06: Inserir produtos até o valor R$990,10, não deve permitir a finalização da compra e exibir uma mensagem de alerta;

CT07: Compras até R$199,99 não devem receber desconto; CT08: Compras de R$200,00 devem receber 10% de desconto; CT09: Compras de R$400,00 devem receber 10% de desconto; CT10: Compras de R$600,00 devem receber 10% de desconto; CT09: Compras de R$601,00 devem receber 15% de desconto.

# História de usuário 2:

CT01: Ao logar com um usuário ativo, com login e senha corretos, deve ser possível visualizar pedidos;

CT02: Ao inserir um login correto e senha incorreta, deve exibir uma mensagem de erro;

CT03: Ao inserir um login incorreto e senha correta, deve exibir uma mensagem de erro;

CT04: Ao inserir um login incorreto e senha incorreta, deve exibir uma mensagem de erro;

CT05: Ao inserir um login incorreto e senha incorreta, deve exibir uma mensagem de erro;

CT06: Ao inserir um login com formato de e-mail e senha correta, deve permitir acesso ao sistema;

CT07: Ao inserir um login de nome e senha correta, deve permitir acesso ao sistema;

CT08: Ao inserir um login com formato de CPF e senha correta, deve permitir acesso ao sistema;

CT09: Ao inserir um login ou senha incorretos por mais de 3 vezes, não deve ser possível fazer outra tentativa pelos próximos 15 minutos.

# História de usuário 3: API de Cupom

CT01: Ao ir no método GET e não passar parâmetro, ao clicar para listar, devem ser retornados todos os cupons cadastrados;

CT02: Ao ir no método GET e passar um ID como parâmetro, ao clicar para listar, ser listado apenas o cupom com o ID informado;

CT03: Ao ir no método POST e informar corretamente o Código do cupom, valor, tipo de desconto e descrição como parâmetros, ao clicar para cadastrar o cupom, ser cadastrado com sucesso;

CT04: Ao ir no método POST e informar corretamente o Código do cupom, valor e tipo de desconto, mas não informar a descrição, ao clicar para cadastrar o cupom, deve ser exibida mensagem de alerta para informar campo obrigatório;

CT05: Ao ir no método POST e informar corretamente o Código do cupom, valor e descrição, mas não informar o tipo de desconto, ao clicar para cadastrar o cupom, deve ser exibida mensagem de alerta para informar campo obrigatório;

CT06: Ao ir no método POST e informar corretamente o Código do cupom, tipo de desconto e descrição, mas não informar o valor, ao clicar para cadastrar o cupom, deve ser exibida mensagem de alerta para informar campo obrigatório;

CT07: Ao ir no método POST e informar corretamente o valor, tipo de desconto e descrição, mas não informar o Código do cupom, ao clicar para cadastrar o cupom, deve ser exibida mensagem de alerta para informar campo obrigatório;

CT08: Ao ir no método POST e tentar cadastrar cupom com nome igual a um já previamente cadastrado, deve ser exibida mensagem de erro informando que não pode repetir o nome.

# Repositório no Github

* Link do repositório: https://github.com/leticia-code/TCC-EBAC

# Testes automatizados

* + 1. Automação de UI
* Crie um projeto de automação no Cypress;
* Crie uma pasta chamada UI para os testes WEB da História de Usuário [US-0001] – Adicionar item ao carrinho;
* Na automação deve adicionar pelo menos 3 produtos diferentes e validar se os itens foram adicionados com sucesso.

# Link da atividade no repositório: https://github.com/leticia-code/TCC-EBAC/tree/main/Atividades%20TCC/UI

* + 1. Automação de API
* Crie uma pasta chamada API para os testes de API da História de usuário **“Api de cupons”**.
* Faça a automação de **listar** os cupons e **cadastrar** cupom, seguindo as regras da História de usuário.

# Link da atividade no repositório: https://github.com/leticia-code/TCC-EBAC/tree/main/Atividades%20TCC/API

# Integração contínua

* Coloque os testes automatizados na integração contínua com jenkins, criando um job para execução da sua automação;
* Compartilhe o *jenkinsfile* no repositório, junto ao seu projeto.

# Link do repositório: https://github.com/leticia-code/TCC-EBAC/tree/main/Atividades%20TCC/API

# Testes de performance

* Usando o Apache Jmeter, faça um teste de performance com o fluxo de login da História de usuário: [US-0002] – Login na plataforma
* Crie um template de gravação no jmeter (recording);
* Use massa de dados dinâmica em arquivo CSV;
* Configurações do teste de performance:

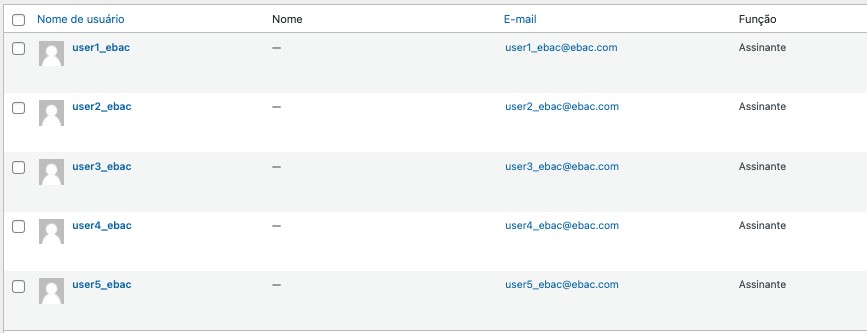
-Usuários virtuais: 20

-Tempo de execução: 2 minutos

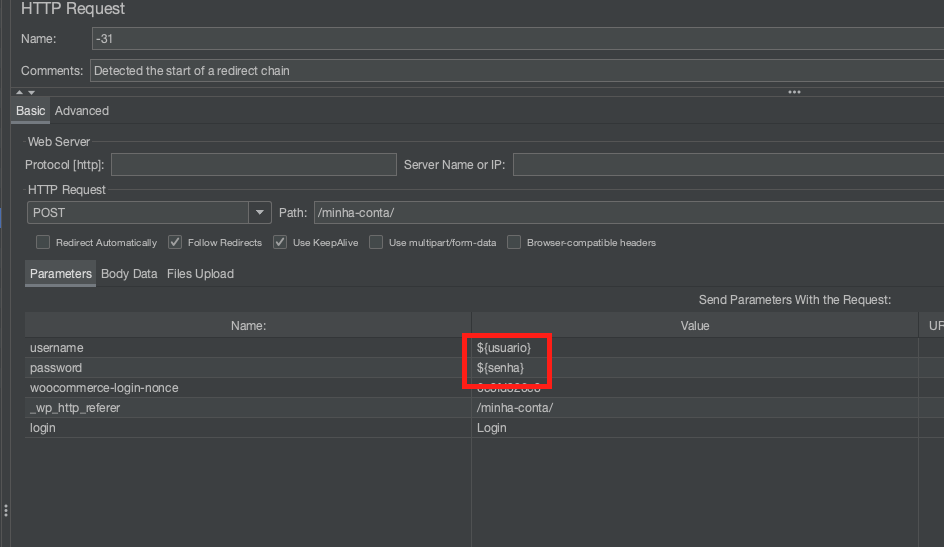
-RampUp: 20 segundos

-Massa de dados: Usuário / senha:

user1\_ebac / psw!ebac@test user2\_ebac / psw!ebac@test user3\_ebac / psw!ebac@test user4\_ebac / psw!ebac@test user5\_ebac / psw!ebac@test



* + DICA: Em uma das requisições, após a gravação, vai aparecer os parâmetros usados. Substitua esses parâmetros pela sua massa de dados, conforme aprendido em aula:



# - Link da atividade no repositório: https://github.com/leticia-code/TCC-EBAC/tree/main/Atividades%20TCC/Teste%20Performance%20TCC

# CONCLUSÃO

Este trabalho de conclusão do Curso de Teste de Software da plataforma EBAC foi um compilado de tudo o que foi estudado ao longo dos capítulos e foi uma oportunidade de reforçar ainda mais os conhecimentos aprendidos. O curso foi acrescentou bastante aos conhecimentos e irá agregar ao trabalho de testes executados no dia-a-dia, com foco aos testes automatizados, que é de grande importância no mercado atual e que é capaz de agilizar os testes. Por fim, foi uma oportunidade de conhecer outros profissionais da área e compartilhar experiências.

# REFERÊNCIAS BIBLIOGRÁFICAS

EBAC – Escola Britânica de Artes Criativas & Tecnologia. **Curso Teste de Software.** Disponível em: https://ebaconline.com.br/qualidade-de-software. Acesso em: 18 fev. 2023.