



## QUALIDADE DE SOFTWARE

Larissa Bravo de Faria Castro de Oliveira

Análise de Qualidade

Ribeirão Preto / SP

2024

## **RESUMO**

Este trabalho apresenta um projeto de conclusão do curso de Qualidade de Software, abordando desde o planejamento à estratégia e execução de testes manuais e automatizados.

## SUMARIO

<b>1. INTRODUÇÃO .....</b>	<b>4</b>
<b>2. PROJETO .....</b>	<b>5</b>
2.1. ESTRATÉGIA DE TESTE .....	5
2.2. CRITÉRIO DE ACEITAÇÃO.....	5
2.2.1. História de usuário 1: [US-0001] – Adicionar item ao carrinho .....	6
2.2.2. História de usuário 2: [US-0002] – Login na plataforma .....	7
2.2.3. História de usuário 2: [US-0003] – API de cupons .....	8
2.3. CASOS DE TESTES .....	8
2.3.1. História de usuário 1: Adicionar item ao carrinho .....	9
2.3.2. História de usuário 2: Login na plataforma.....	9
2.3.3. História de usuário 3: API de Cupom .....	10
2.4. REPOSITÓRIO NO GITHUB.....	10
2.5. TESTES AUTOMATIZADOS.....	10
2.5.1. Automação de UI .....	10
2.5.2. Automação de API.....	11
2.5.3. Integração contínua .....	11
2.5.4. Testes de performance .....	12
<b>3. CONCLUSÃO .....</b>	<b>14</b>
<b>REFERÊNCIA .....</b>	<b>15</b>

## **1. INTRODUÇÃO**

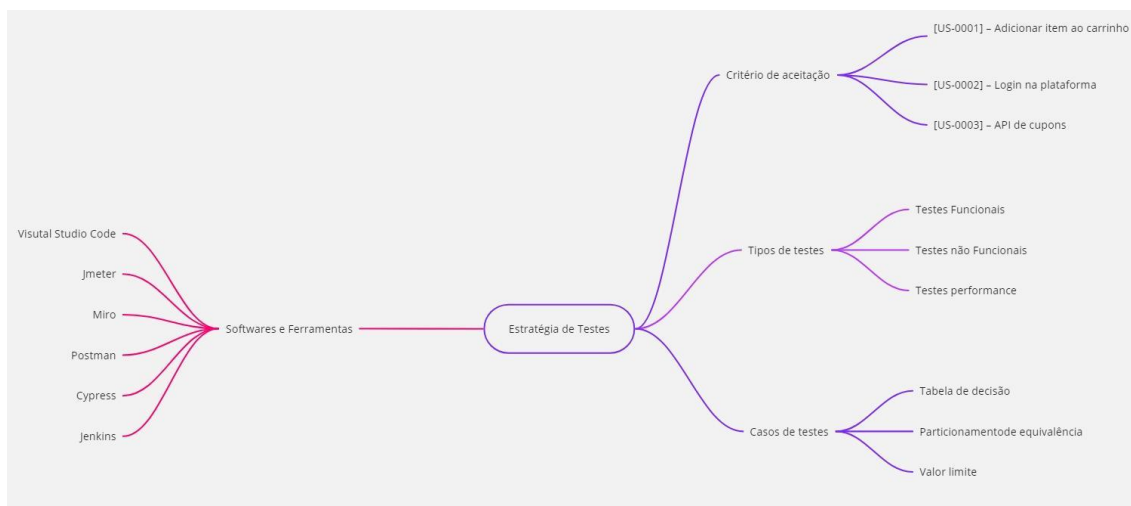
O foco central do projeto de conclusão de curso é a análise de requisitos e a implementação de estratégias de testes na loja virtual EBAC Shop, visando assegurar a qualidade do e-commerce em questão. As estratégias utilizadas foram adquiridas ao longo do curso e aplicadas de forma específica no desenvolvimento do projeto.

## 2. O PROJETO

Para o Trabalho de Conclusão de Curso Qualidade de Software, será considerado as histórias de usuário refinadas, como se estivesse participando de um time ágil. As funcionalidades devem seguir todo o fluxo de trabalho de um QA, desde o planejamento até a entrega. Siga as etapas dos sub-tópicos para te orientar no trabalho. Todas as boas práticas, tanto de documentação, escrita e desenvolvimento, serão consideradas na nota. Portanto caprichem, pois além de trabalho servir como nota para o curso, vai servir como Portfólio em seu github.

### 2.1. ESTRATÉGIA DE TESTE

- Faça uma estratégia de testes em um mapa mental, seguindo algumas diretrizes como objetivos, papéis e responsabilidades, fases de testes, padrões, tipos de testes, técnicas de testes, ambientes, ferramentas, abordagem (manual ou automatizado), framework ou ferramenta usados, plataformas (web, api, mobile), etc.;
- Referência: Módulo 5
- Após fazer sua estratégia de teste, tire um print e cole aqui:



### 2.2. CRITÉRIOS DE ACEITAÇÃO

- Considere as histórias de usuário: [US-0001] – Adicionar item ao carrinho, [US-0002] – Login na plataforma e [US-0003] – API de cupons

- Para cada uma delas crie pelo menos 2 critérios de aceitação usando a linguagem Gherkin;
- Em pelo menos um dos critérios, usar tabela de exemplos ( Esquema do Cenário / Scenario Outline);
- Referência: Módulo 8

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

**Como** cliente EBAC Shop

**Quero** adicionar produtos no carrinho

**Para** realizar a compra dos itens

Critérios de aceitação:

- 1 - Deve existir uma opção para adicionar mais quantidades de itens do produto no carrinho de compras;
- 2 - Deve existir uma opção para remover os itens do produto no carrinho de compras;
- 3 - Deve existir uma página com todos os itens adicionados no carrinho de compras;
- 4 - Deve existir uma opção para atualizar o carrinho de compras.

Cenário 1: Adicionar item ao carrinho com sucesso

**Dado** que usuário está no site EBAC Shop

**E** visualizando a página de produtos

**Quando** clicar no botão 'Comprar'

**Então** o item deve ser adicionado ao carrinho de compras

**E** exibir a mensagem 'Item adicionado ao carrinho'

Cenário 2: Acessar carrinho de compras e adicionar mais item

**Dado** que usuário está no site EBAC Shop

**E** visualizando a página de produtos

**Quando** clicar no botão 'Comprar'

**E** clicar no botão 'Ver Carrinho'

**E** clicar no botão '+' para adicionar mais itens do produto

**Então** a quantidade do item deve ser atualizada automaticamente.

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

**Como** cliente da EBAC-SHOP

**Quero** fazer login (autenticação) na plataforma

**Para** visualizar meus pedidos

Critérios de aceitação:

- 1 - Para autenticação na plataforma, o cliente deve ter cadastro;
- 2 - Deve existir uma lista de pedidos do usuário;
- 3 - Deve Apresentar todos os dados do pedido;
- 4 - Para acompanhar os pedidos em aberto, deve existir o status do pedido.

Cenário 1: Login com usuário e senha válidos

**Dado** que usuário está no site EBAC Shop

**Quando** clicar em login

**E** informar usuário e senha válidos

**Então** deve ser redirecionado o painel Minha Conta

Cenário 2: Tentativa de login

**Dado** que o usuário está no site EBAC Shop

**Quando** clicar no botão login

**E** informar o nome de usuário <username>

**E** informar a senha <password>

**E** clicar no botão Login

**Então** deve ser exibido o texto <message>.

Exemplos:

| username | password | message

| usuario\_valido | senha123 | Bem-vindo <username>

| usuario\_invalido | senha456 | O usuário <username> não está registrado no

site

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

**Como** gerente do site da EBAC-SHOP

**Quero** uma API de Cupons

**Para** oferecer descontos aos clientes de forma fácil e personalizada.

Critérios de aceitação:

- 1 - A API deve ser segura e requerer autenticação para todas as operações;
- 2 - A API deve fornecer mensagens de erro claras e informativas em caso de problemas;
- 3 - Os cupons expirados ou atingidos pelo limite de uso não devem ser aplicáveis aos carrinhos de compras.

Cenário 1: Criar um novo cupom

**Dado** que tenha acesso à API de Cupons

**Quando** enviar uma Requisição POST para criar um novo cupom

**Então** a API deve retornar status de sucesso (200 ok) e retornar o id do novo cupom

Cenário 2: Aplicar cupom a um carrinho de compras

**Dado** que o usuário tenha um carrinho de compras com produto

**E** o cliente tenha um código de cupom válido

**Quando** o cliente aplicar o cupom ao carrinho

**Então** o valor total do carrinho deve ser recalculado levando em consideração o desconto do cupom

### 2.3. CASOS DE TESTES

- Crie pelo menos 3 casos de testes para cada história de usuário, sempre que possível, usando as técnicas de testes (partição de equivalência, valor limite, tabela de decisão etc.).
- Considere sempre o caminho feliz (fluxo principal) e o caminho alternativo e negativo (fluxo alternativo). Exemplo de cenário negativo: “Ao preencher com usuário e senha inválidos deve exibir uma mensagem de alerta...”
- Referência: Módulo 4 e 5



### 2.3.1. História de usuário 1: Adicionar item ao carrinho

**Como** cliente da EBAC Shop

**Quero** adicionar produtos no carrinho

**Para** realizar compras dos itens

#### Regra de negócio:

CT01: Não é permitido inserir mais de 10 itens de um mesmo produto no carrinho;

CT02: Os valores não podem ultrapassar de R\$ 999,00;

Regra	Entrada	Saída
CT01	Inserir 5 itens do produto Josie Yoga Jacket	Válida
CT01	Inserir 15 itens do produto Augusta Pullover Jacket	Inválida
CT02	Itens do carrinho totalizando no valor de R\$ 2.000,00	Inválida
CT02	Itens do carrinho totalizando no valor de R\$ 500,00	Válida

### 2.3.2. História de usuário 2: Login na plataforma

CT01: Somente usuários ativos podem fazer login; (tabela de decisão)

CT02: Deve exibir uma mensagem de erro caso usuário erre o login e senha;

CT03: Login deve permitir e-mail, nome de usuário ou cpf;

Condições	Regra 1	Regra 2	Regra 3	Regra 4
Usuário válido?	Sim	Sim	Não	Não
Usuário ativo?	Sim	Sim	Não	Não
Senha válida?	Sim	Não	Sim	Não
Ações				
Permitir Acesso?	Sim	Não	Não	Não
Exibir mensagem?	Não	Sim	Sim	Sim

### 2.3.3. História de usuário 3: API de Cupom

CT01: Deve permitir cadastrar um novo cupom apenas autenticado;

CT02: Deve listar todos os cupons cadastrados ou listar buscando por ID do cupom;

CT03: Não deve permitir cadastro de cupom duplicado.

## 2.4. REPOSITÓRIO NO GITHUB

- Crie um repositório no github com o nome TCC-EBAC;
- Deixe o repositório publico até a análise dos tutores;
- Neste repositório você deve subir este arquivo e todos os código fontes da automação WEB, API, Mobile, Performance e CI.
- Referência: Módulo 10
- Link do repositório: <https://github.com/lariibravo/TCC-EBAC>

## 2.5. TESTES AUTOMATIZADOS

### 2.5.1. 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.

### 2.5.2. 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.
- Exemplo da automação de Api – GET

```
it('Deve listar todos os cupons cadastrados', () => {
  cy.request({
    method: 'GET',
    url: 'coupons',
    headers: {
      authorization: 'código_da_autorização_aqui'
    }
  }).should((response) => {
    cy.log(response)
    expect(response.status).to.equal(200)
  })
});
```

- Obs.: Considere todas as boas práticas de otimização de cenários (Page Objects, Massa de dados, Custom Commands, elementos etc.).
- Referência: Módulo 11, 12 e 14






### 2.5.3. 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.
- Referência: Módulo 15

#### 2.5.4. 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;
- Referência: Módulo 18
- 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

<input type="checkbox"/> Nome de usuário	Nome	E-mail	Função
<input type="checkbox"/>  user1_ebac	—	user1_ebac@ebac.com	Assinante
<input type="checkbox"/>  user2_ebac	—	user2_ebac@ebac.com	Assinante
<input type="checkbox"/>  user3_ebac	—	user3_ebac@ebac.com	Assinante
<input type="checkbox"/>  user4_ebac	—	user4_ebac@ebac.com	Assinante
<input type="checkbox"/>  user5_ebac	—	user5_ebac@ebac.com	Assinante

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

HTTP Request

Name: -31

Comments: Detected the start of a redirect chain

Basic Advanced

Web Server

Protocol [http]: Server Name or IP:

HTTP Request

POST Path: /minha-corta/

☐ Redirect Automatically ☒ Follow Redirects ☒ Use KeepAlive ☐ Use multipart/form-data ☐ Browser-compatible headers

Parameters Body Data Files Upload

Send Parameters With the Request:

Name:	Value	UF
username	\$(usuario)	
password	\$(senha)	
woocommerce-login-nonce	6c67a2c03e0	
_wp_http_referer	/minha-corta/	
login	Login	

### **3. CONCLUSÃO**

A conclusão com o projeto em questão reflete a materialização do conhecimento adquirido ao longo do curso de Qualidade de Software.

Todo o projeto foi aplicado à metodologia ágeis, no formato do dia a dia de um QA em ação e isto trouxe experiência para certificação e futuros projetos.

## **REFERÊNCIAS BIBLIOGRÁFICAS**

**EBAC:**<https://ebaonline.com.br/>