



QUALIDADE DE SOFTWARE

Rayane Oliveira

Análise de Qualidade

Araxá - MG

2025

1. RESUMO

Este trabalho tem como objetivo demonstrar, na prática, os principais conceitos e técnicas da área de Qualidade de Software apresentados ao longo do curso da EBAC. Foram explorados temas como planejamento de testes, critérios de aceitação, testes funcionais e não funcionais, testes automatizados, testes mobile e de API, além da integração contínua e testes de performance. As atividades foram organizadas com base em histórias de usuário simulando o contexto de um time ágil. Para isso, foram utilizadas ferramentas como Cypress, Postman, Swagger, Docker, Jenkins, Appium, Android Studio, SauceLabs, JMeter, DBeaver, MongoDB, Git e GitHub. A aplicação prática desses conceitos mostrou como a adoção de uma abordagem de qualidade desde o início do ciclo de desenvolvimento contribui para a entrega de softwares mais confiáveis e com menor risco de falhas.

2. SUMÁRIO

1. RESUMO	2
2. SUMÁRIO	3
3. INTRODUÇÃO	4
4. O PROJETO	5
4.1 Estratégia de teste	5
4.2 Critérios de aceitação.....	6
4.2.1 História de usuário 1: [US-0001] – Adicionar item ao carrinho	6
4.2.2 História de usuário 2: [US-0002] – Login na plataforma	6
4.2.3 História de usuário 2: [US-0003] – API de cupons.....	7
4.3 Casos de testes	8
4.3.1 História de usuário 1:	8
4.3.2 História de usuário 2:	8
4.3.1 História de usuário 3: API de Cupom	8
4.4 Repositório no Github.....	9
4.5 Testes automatizados	9
4.6 Integração contínua	10
4.7 Testes de performance.....	10
5. CONCLUSÃO	11
6. REFERÊNCIAS BIBLIOGRÁFICAS.....	11

3. INTRODUÇÃO

A qualidade de software é uma área essencial dentro do desenvolvimento de sistemas, pois garante que as aplicações sejam entregues com o mínimo de erros, dentro dos requisitos estabelecidos e com boa experiência para o usuário. Ao longo do curso de Qualidade de Software da EBAC, foram apresentados diversos tópicos que fazem parte do dia a dia de um profissional de QA. Este trabalho reúne e aplica esses conhecimentos por meio de histórias de usuário, simulando o fluxo de trabalho de um time ágil. Serão demonstradas etapas como planejamento, definição de critérios de aceitação, criação de casos de teste, automação, testes de performance e integração contínua. O objetivo é consolidar os conhecimentos adquiridos, aplicando as melhores práticas e ferramentas para garantir a qualidade do software.

4. O PROJETO

Para o Trabalho de Conclusão de Curso Qualidade de Software, você deve considerar as histórias de usuário já refinadas e como se você 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 subtópicos 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.

4.1 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:



[Imagem: Mapa mental – Estratégia de teste]

4.2 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

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

Critérios de aceitação:

Contexto:

Dado que eu acesse a plataforma da Ebac-Shop

Cenário 1: Quantidade de produtos por venda

Quando eu adicionar um produto no carrinho

E adicionar a <quantidade>

Então deve exibir a <mensagem>

Exemplo:

quantidade	mensagem
8	"Quantidade adicionada com sucesso"
10	"Quantidade adicionada com sucesso"
12	"Quantidade excedida, valor permitido de apenas 10 produtos por venda"

Cenário 2: Valor da compra para ganhar cupom

Quando eu adicionar produtos no carrinho

E o valor da minha compra for maior que R\$200,00

Então devo ganhar um cupom de desconto

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

Critérios de aceitação:

Contexto:

Dado que eu acesse a página de login na plataforma da Ebac-Shop

Cenário 1: Login com sucesso

Quando eu digitar um usuário e senha válidos

E clicar no botão de login

Então deve fazer login com sucesso e redirecionar para a página Minha conta

Cenário 2: Login com usuário e senha inválidos

Quando eu digitar um usuário e senha inválidos

E clicar no botão de login

Então deve exibir a mensagem de erro “Usuário ou senha inválidos”

4.5 História de usuário 3: [US-0003] – API de cupons

Critérios de aceitação:

Contexto:

Dado que eu acesse a API de cupons como administrador devidamente autenticado

Cenário 1: Listar cupons cadastrados

Quando enviar a requisição GET para visualizar cupons

Então deve mostrar todos os cupons cadastrados e retornar o Status Code 200

Cenário 2: Cadastrar cupom

Quando na requisição POST preencher os dados necessários para cadastro: Código do cupom, valor, tipo de desconto, descrição

E enviar a requisição

Então deve cadastrar cupom com sucesso e retornar o Status Code 200

4.6 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

4.7 História de usuário 1: Adicionar item ao carrinho

CT01: Inserir mais de 10 itens de um mesmo produto no carrinho

Deve exibir mensagem: “Quantidade excedida”

Técnica: Valor limite

CT02: Realizar compra com valor entre R\$200,00 e R\$600,00

Deve ganhar cupom de 10% de desconto

Técnica: Partição de equivalência

CT03: Realizar compra com valor acima de R\$600,00

Deve ganhar cupom de 15% de desconto

Técnica: Partição de equivalência

4.8 História de usuário 2: Login na plataforma

CT01: Realizar login com usuário e senha válidos

Deve permitir acesso

Técnica: Fluxo principal/ Caminho feliz

CT02: Realizar login com usuário e senha inválidos

Deve exibir mensagem de erro: “Usuário ou senha inválidos”

Técnica: Fluxo alternativo

CT03: Errar a senha 3 vezes seguidas

Deve travar o login por 15 minutos

Técnica: Fluxo negativo/ Cenário de segurança

4.8.1 História de usuário 3: API de Cupom

CT01: Tentar cadastrar cupom com nome repetido

Deve exibir erro

Técnica: Fluxo negativo

CT02: Cadastrar cupom com todos os campos obrigatórios preenchidos corretamente

Deve permitir cadastro e retornar Status code 200

Técnica: Fluxo principal/ Caminho feliz

CT03: Fazer listagem e cadastro de cupons sem autenticação de administrador

Deve retornar erro de permissão

Técnica: Fluxo negativo/ Controle de acesso/ Autenticação

4.9 Repositório no Github

- Crie um repositório no github com o nome TCC-EBAC;
- Deixe o repositório público 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/rm-oliveira/TCC-EBAC>

4.10 Testes automatizados

4.10.1 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.

4.10.2 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

4.11 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

4.12 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:

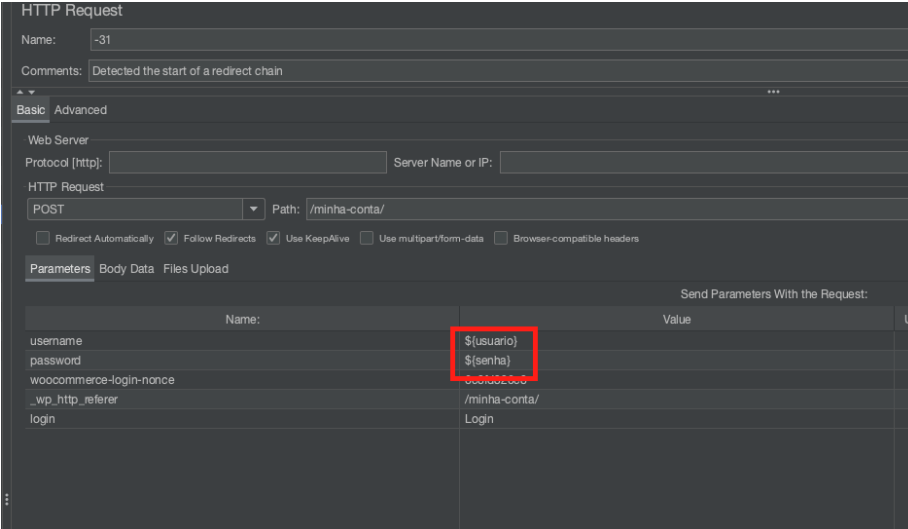
```

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 usados. Substitua esses parâmetros pela sua massa de dados, conforme aprendido em aula:



The screenshot shows an HTTP Request tool interface. The 'Parameters' tab is selected, displaying a table of request parameters. The 'username' parameter has a value of '\$(usuario)' and the 'password' parameter has a value of '\$(senha)'. Both values are highlighted with a red box. Other parameters include 'woocommerce-login-nonce', '_wp_http_referer', and 'login'.

Name:	Value
username	\$(usuario)
password	\$(senha)
woocommerce-login-nonce	0000000000
_wp_http_referer	/minha-conta/
login	Login

5. CONCLUSÃO

Ao longo do trabalho foi possível perceber como o conjunto de práticas e ferramentas abordadas no curso de Qualidade de Software da EBAC se complementam e contribuem para um processo de desenvolvimento mais robusto e seguro. O uso de testes automatizados e práticas de DevOps reduz falhas, melhora a velocidade de entrega e aumenta a confiança no produto. A aplicação prática reforça que a qualidade deve ser pensada desde o início do desenvolvimento, e não apenas como uma etapa final.

6. REFERÊNCIAS BIBLIOGRÁFICAS

Seguir regras ABNT