



ENGENHEIRO DE QUALIDADE DE SOFTWARE

Luiz Gustavo Coutinho Carvalho

Análise de Qualidade

Itajubá - MG

2023

1. RESUMO

O presente trabalho de conclusão do curso Engenheiro de Qualidade de Software, é dividido em duas abordagens distintas: a primeira consiste em escrita, presente neste arquivo, e a segunda em códigos que, encontraremos no repositório remoto do GitHub.

Na parte escrita serão abordados alguns assuntos, como: Estratégia de Testes, Critérios de Aceitação, Cenários (ou Casos) de Testes, dentre outros temas. Se tratando da parte de códigos, foi criado um repositório com projetos distintos, separados por pastas, abordando os seguintes temas: UI (User Interface), API, Mobile e Performance.

2. SUMÁRIO

1. RESUMO.....	2
2. SUMÁRIO	3
3. INTRODUÇÃO	4
4. O PROJETO	4
4.1 Estratégia de teste	5
4.2 Critérios de aceitação.....	6
4.3 Casos de testes	14
4.4 Repositório no Github	16
4.5 Testes automatizados	17
4.6 Integração contínua	19
4.7 Testes de performance.....	19
5. CONCLUSÃO	20
6. REFERÊNCIAS BIBLIOGRÁFICAS.....	21

3. INTRODUÇÃO

O profissional de Qualidade de Software, especialmente o Engenheiro de Qualidade, participa de todo o ciclo de desenvolvimento do software, desde o refinamento de um item do backlog, até a entrega para a homologação do cliente, ou até mesmo para usuário final. Sendo assim, este trabalho também irá tratar de assuntos que envolvem diferentes etapas da concepção de um software.

Começaremos com a estratégia dos testes, onde será apresentado em formato de mapa mental, as plataformas a que submeteremos o nosso software aos testes, as tecnologias envolvidas, os níveis dos testes, dentre outros.

Relacionados à documentação, será visto no presente trabalho os Critérios de Aceitação de algumas histórias de usuário, os Cenários de Testes dessas mesmas histórias, onde será apresentado quais serão automatizados, e também a escrita de algumas Histórias de Usuários.

Por último abordaremos a parte de código, onde será possível ter acesso aos projetos que, resumidamente, utilizam das seguintes tecnologias/ferramentas:

- UI: JavaScript, Cypress, Cucumber;
- API: JavaScript, Supertest e Joi para os testes de contrato;
- Mobile: JavaScript, Appium, Webdriver IO e BrowserStack como Device Farm;
- Performance: JavaScript e k6.io.

4. O PROJETO

Para este trabalho de conclusão de curso **Profissão: Engenheiro de Qualidade de software**, você deve utilizar o conhecimento adquirido ao longo do curso para elaborar uma estratégia de testes adequada para validar o e-commerce EBAC Shop (<http://lojaebac.ebaconline.art.br/>). Você deve considerar as histórias de usuário já refinadas como se você estivesse participando de um time ágil. As funcionalidades devem seguir todo o fluxo de trabalho de um *Quality Engineer*

(QE), desde o planejamento até a entrega. Siga as etapas dos sub-tópicos para se orientar no trabalho.

ATENÇÃO:

- Conforme a sua estratégia, você pode executar os testes no endereço disponibilizado ou utilizando as imagens disponíveis no Docker Hub:
 - Banco de Dados: [ernestosbarbosa/lojaebacdb](https://hub.docker.com/r/ernestosbarbosa/lojaebacdb)
 - Loja EBAC: [ernestosbarbosa/lojaebac](https://hub.docker.com/r/ernestosbarbosa/lojaebac)
- Comandos para subir os containers:

```
docker network create --attachable ebac-network  
  
docker run -d --name wp_db -p 3306:3306 --network ebac-network ernestosbarbosa/lojaebacdb:latest  
  
docker run -d --name wp -p 80:80 --network ebac-network ernestosbarbosa/lojaebac:latest
```

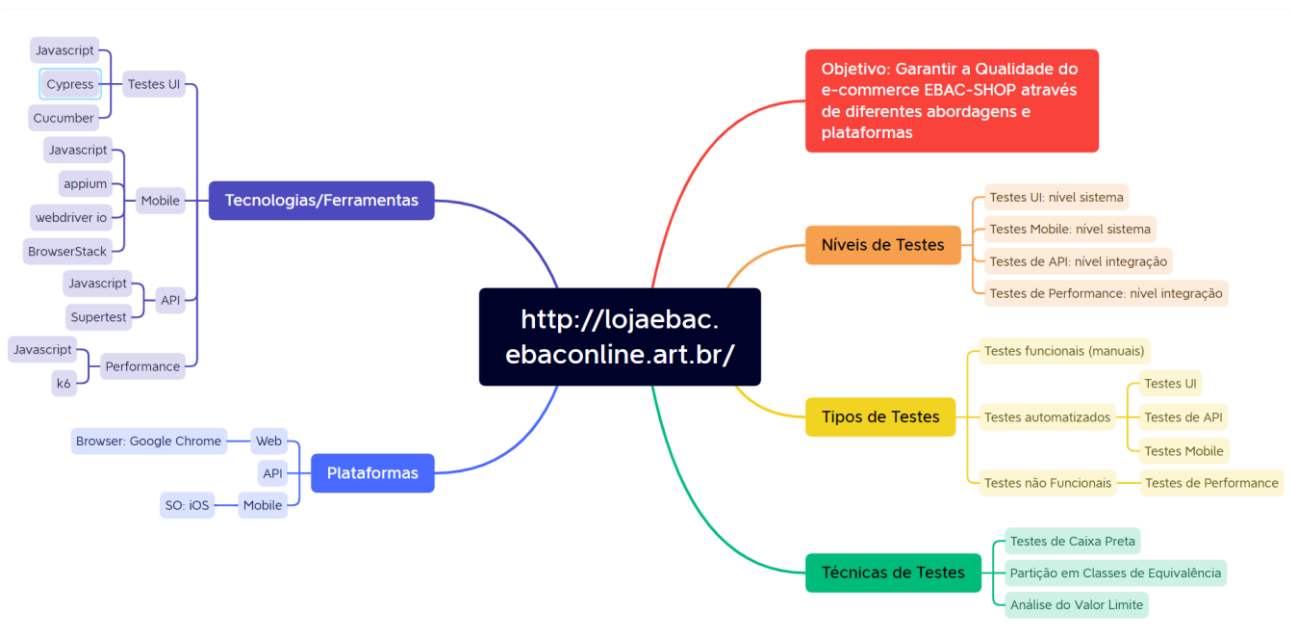
Após subir os containers a loja estará em <http://localhost:80>

- Como este trabalho complementa o que criou em seu Trabalho de Consolidação (Módulo 19), você pode utilizá-lo como base para o seu Trabalho de Conclusão.

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:



4.2 Critérios de aceitação

- Considere as histórias de usuário:
 - [US-0001] – Adicionar item ao carrinho

```

Dado que eu esteja na página do carrinho
Quando eu tentar inserir mais do que 10 itens de um mesmo produto no carrinho
Então me será apresentado um alerta me informando sobre essa regra
E não conseguirei prosseguir para o checkout

Dado que eu esteja na página do carrinho
Quando os valores ultrapassarem R$ 990,00
Então me será apresentado um alerta me informando sobre essa regra
E não conseguirei prosseguir para o checkout

Dado que eu esteja na página do carrinho
Quando os valores estiverem entre R$ 200,00 e R$ 600,00
Então ganharei um cupom de 10% de desconto

Dado que eu esteja na página do carrinho
Quando os valores ultrapassarem R$ 600,00
Então ganharei um cupom de 15% de desconto
  
```

- [US-0002] – Login na plataforma

```

Dado que eu seja um usuário ativo
Quando eu acessar a página de Login
E inserir minhas credenciais corretamente
  
```

```
E submeter
Então farei login com sucesso

Dado que eu seja um usuário com conta inativa
Quando eu acessar a página de Login
E inserir minhas credenciais
E submeter
Então o sistema me apresentará um alerta
E eu não conseguirei logar

Dado que eu seja um usuário ativo
Quando eu acessar a página de Login
E inserir informações incorretas
E submeter
Então o sistema me apresentará um alerta
E eu não conseguirei logar até que eu informe dados corretos

Dado que eu seja um usuário ativo
Quando eu acessar a página de Login
E submeter o login por 3 vezes com dados incorretos
Então o sistema me apresentará um alerta
E eu só poderei tentar logar novamente após passados 15 minutos
```

- [US-0003] – API de cupons

```
Dado que eu tenha me autenticado na API de cupons
Quando eu executar o método GET para /wc/v3/coupons
Então a API deverá listar todos os cupons cadastrados

Dado que eu tenha me autenticado na API de cupons
Quando eu executar o método GET para /wc/v3/coupons/{id}
Então a API deverá listar o produto correspondente ao id informado no path

Dado que eu tenha me autenticado na API de cupons
Quando eu executar o método POST para /wc/v3/coupons, passando todos os campos obrigatórios
Então a API retornará o status code 200, indicando sucesso na operação

Dado que eu tenha me autenticado na API de cupons
Quando eu executar o método POST para /wc/v3/coupons, passando um nome repetido de cupom
Então a API retornará o status code 400, indicando falha na operação
```

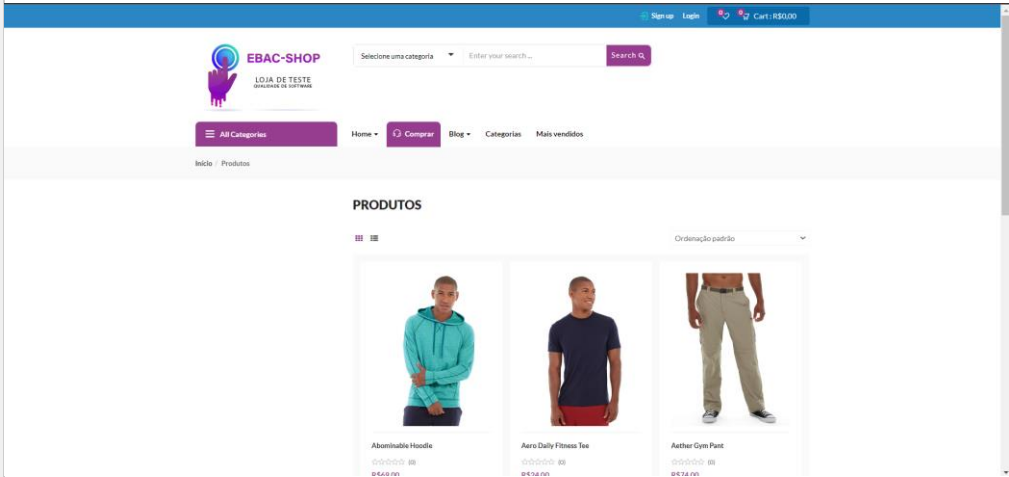
- Para cada uma delas crie pelo menos 4 critérios de aceitação usando a linguagem Gherkin;
- Crie histórias de usuário para as funcionalidades:
 - Catálogo de Produtos

[US-0004] – Catálogo de Produtos

Situação:	Em Andamento
Projeto:	EBAC-SHOP
Versões Afetadas:	1.0

Tipo:	História	Prioridade:	Alta
Solicitante:	Fábio Araújo	Responsável:	
Resolução:		Pontuação:	13
Tempo Estimado:			

Anexos:



The screenshot displays the EBAC-SHOP website interface. At the top, there is a blue navigation bar with links for 'Sign up', 'Login', and a shopping cart icon showing 'Cart: R\$0,00'. Below this is a white header area with the EBAC-SHOP logo, a search bar with the placeholder 'Selecione uma categoria' and 'Enter your search...', and a 'Search' button. A purple navigation bar contains links for 'All Categories', 'Home', 'Comprar', 'Blog', 'Categorias', and 'Mais vendidos'. The main content area is titled 'PRODUTOS' and shows a grid of three product cards. Each card features a product image, the product name, and the price. The products shown are 'Abominable Hoodie' (R\$450,00), 'Aero Daily Fitness Tee' (R\$120,00), and 'Auther Gym Pant' (R\$120,00). A dropdown menu for 'Ordenação padrão' is visible above the product grid.

Descrição

Como cliente da EBAC-SHOP
Quero visualizar uma lista de produtos
Para escolhê-los em uma compra online

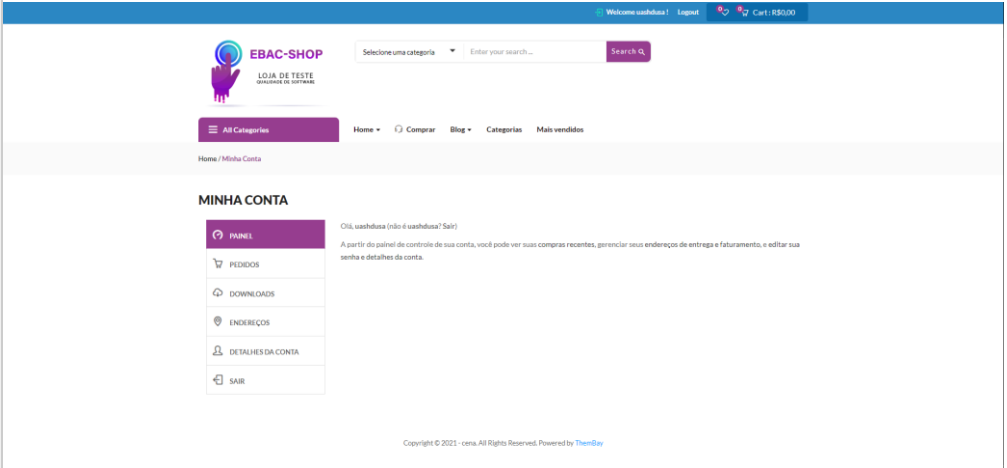
Critérios de Aceitação:

- 1 – Serão apresentados 9 cards de produtos por página
- 2 – Deve apresentar no card de cada produto: nome avaliação do produto e valor unitário

○ Painel Minha Conta

[US-0005] – Painel Minha Conta	
Situação:	Em Andamento
Projeto:	EBAC-SHOP
Versões Afetadas:	1.0

Tipo:	História	Prioridade:	Alta
Solicitante:	Fábio Araújo	Responsável:	
Resolução:		Pontuação:	8
Tempo Estimado:			

Anexos:	
----------------	--

Descrição	<p>Como cliente da EBAC-SHOP Quero acessar minha área logada Para ter acesso aos meus dados cadastrais</p>
------------------	---

Critérios de Aceitação:

- 1 – Deve apresentar mensagem de boas-vindas com o nome de usuário do cliente
- 2 – Deve apresentar link possibilitando ao cliente fazer Logout

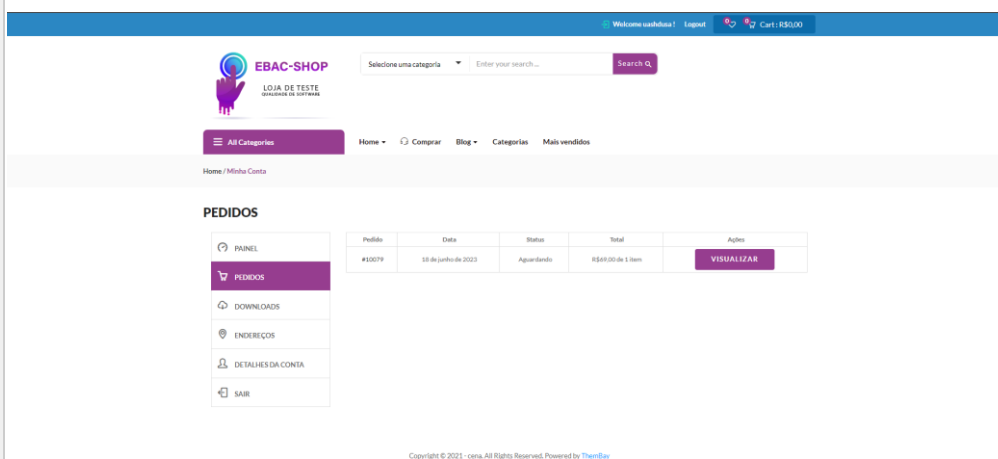
○ Meus Pedidos

[US-0006] – Meus Pedidos

Situação:	Em Andamento
Projeto:	EBAC-SHOP
Versões Afetadas:	1.0

Tipo:	História	Prioridade:	Média
Solicitante:	Fábio Araújo	Responsável:	
Resolução:		Pontuação:	8
Tempo Estimado:			

Anexos:



Descrição

Como cliente da EBAC-SHOP
Quero acessar o menu Pedidos, em Minha Conta
Para visualizar meu histórico de compras na plataforma

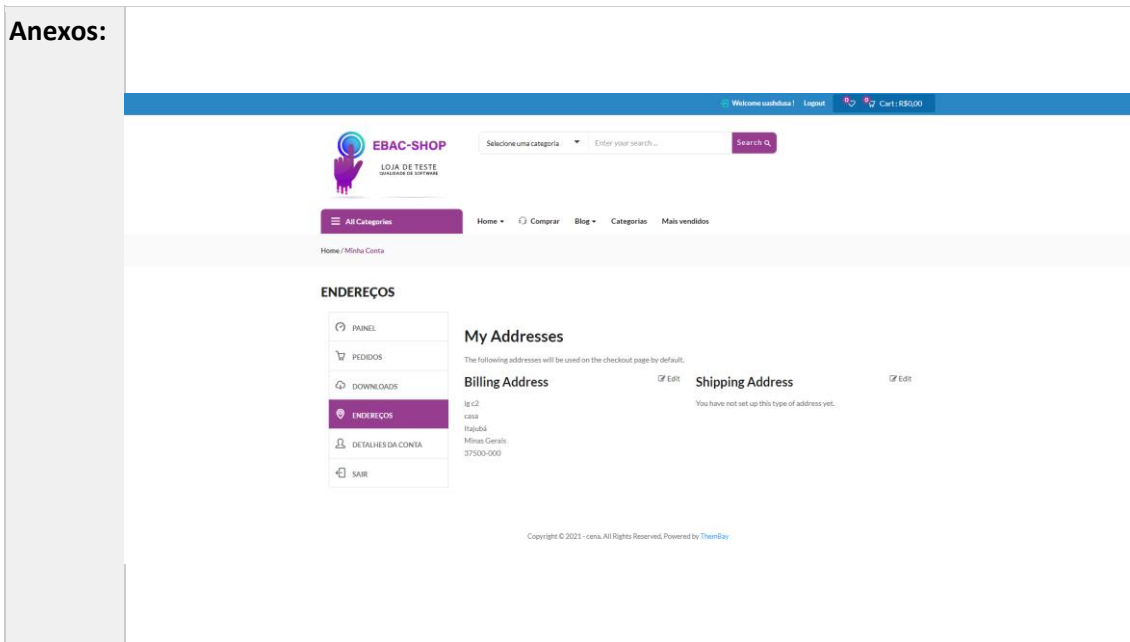
Critérios de Aceitação:

- 1 – Deve ser um menu da página Minha Conta
- 2 – Deve apresentar uma tabela contendo as seguintes colunas: Pedido, Data, Status, Total e Ações
- 3 – Na coluna Ações deve apresentar um botão onde, ao ser clicado, apresentará os detalhes do pedido

○ Endereços

[US-0007] – Endereços	
Situação:	Em Andamento
Projeto:	EBAC-SHOP
Versões Afetadas:	1.0

Tipo:	História	Prioridade:	Média
Solicitante:	Fábio Araújo	Responsável:	
Resolução:		Pontuação:	8
Tempo Estimado:			



Descrição

Como cliente da EBAC-SHOP
Quero acessar o menu Endereços, em Minha Conta
Para visualizar e editar meus endereços de cobrança e de envio

Critérios de Aceitação:

- 1 – Deve ser um menu da página Minha Conta
- 2 – Deve apresentar uma coluna para o endereço de cobrança e outra para o endereço de entrega
- 3 – Deve possibilitar ao usuário editar qualquer um dos endereços

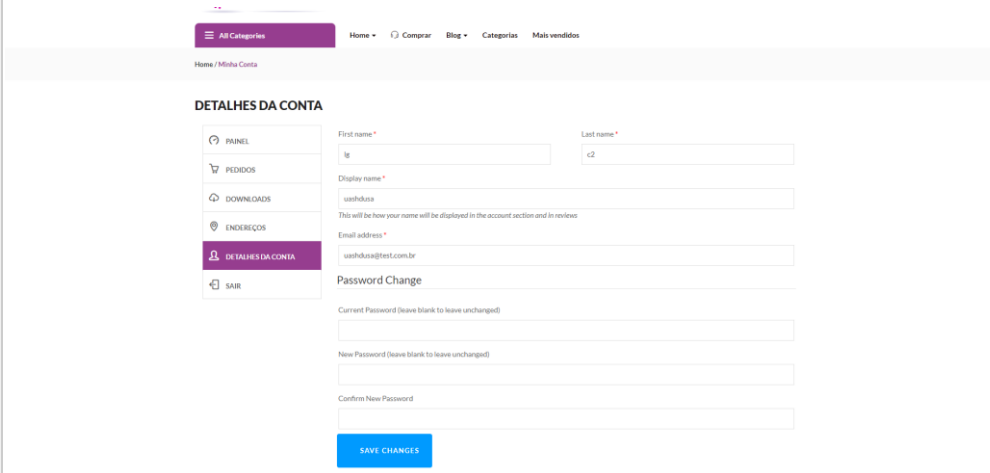
- Detalhes da Conta

[US-0008] – Detalhes da Conta	
Situação:	Em Andamento
Projeto:	EBAC-SHOP
Versões Afetadas:	1.0

Tipo:	História	Prioridade:	Alta
Solicitante:	Fábio Araújo	Responsável:	

Resolução:		Pontuação:	8
Tempo Estimado:			

Anexos:



Descrição

Como cliente da EBAC-SHOP
Quero acessar o menu Detalhes da Conta, em Minha Conta
Para visualizar e editar meus dados de usuário

Critérios de Aceitação:

- 1 – Deve ser um menu da página Minha Conta
- 2 – Na primeira seção, deve possibilitar a edição dos campos: First name, Last name, Display name e Email address
- 3 – Na seção de senha, deve possibilitar alterá-la, através dos seguintes campos: Current Password, New Password e Confirm New Password

- Referência: Módulo 8

4.3 Casos de testes

- Crie pelo menos 4 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...”
- Identifique quais os casos de teste serão automatizados, sendo ao menos 1 caminho feliz e 1 caminho alternativo.
- Referência: Módulos 4 e 5

[US-0001] – Adicionar item ao carrinho

Contexto:

Dado que eu esteja na página do carrinho

@automatizado

Cenário: Validar inserção de menos de 11 itens de um mesmo produto ao carrinho

Quando eu inserir 10 itens de um mesmo produto no carrinho

Então a quantidade apresentada na coluna “Quantity” será atualizada

E os valores também serão atualizados de acordo com a nova quantidade de itens informada

@automatizado

Cenário: Validar valor sem direito a cupom de desconto

Quando eu inserir itens ao carrinho, totalizando um valor menor do que R\$200,00

Então o sistema não me dará um cupom de desconto

E verei o valor cheio na página do carrinho

@manual

Cenário: Validar inserção de 11 itens de um mesmo produto ao carrinho

Quando eu inserir 11 itens de um mesmo produto no carrinho

Então a quantidade apresentada na coluna “Quantity” será atualizada

E receberei um alerta informando sobre a impossibilidade desta ação

E o botão “Quantity” ficará desabilitado, impossibilitando que eu passe para o checkout

@manual

Esquema do Cenário: Validar direito ao cupom de 10%

Quando eu inserir itens ao carrinho, e o valor se atualizar para <valor>

Então ganharei um cupom de 10%

Exemplos:

| valor |

| 200,00 |

| 201,00 |

| 599,00 |

| 600,00 |

@manual

Esquema do Cenário: Validar direito ao cupom de 15%

Quando eu inserir itens ao carrinho, e o valor se atualizar para <valor>

Então ganharei um cupom de 15%

Exemplos:

| valor |

| 600,01 |

| 601,00 |

[US-0002] – Login na plataforma

@automatizado

Cenário: Fazer login

Dado que acesse o site da EBAC-SHOP

E posteriormente a página de login

Quando eu inserir um usuário válido

E a senha correta

Então farei login com sucesso

E serei redirecionado para a tela `minha-conta`

@automatizado

Esquema do Cenário: Tentar realizar login com dados inválidos

Dado que acesse o site da EBAC-SHOP

E posteriormente a página de login

Quando eu inserir o usuário <usuario>

E a senha <senha>

Então me será apresentada a mensagem: <mensagem>

Exemplos:

usuario	senha	mensagem	
usuarioinvalido@iiasuhiauh.com.br	senhaCorreta123	Usuário ou senha inválidos	
usuariovalido@ebac.com.br	senhaIncorreta123	Usuário ou senha inválidos	
usuarioinvalido@iiasuhiauh.com.br	senhaIncorreta123	Usuário ou senha inválidos	

@manual

Cenário: Tentar logar com usuário inativo

Dado que eu seja um usuário com conta inativa

Quando eu acessar a página de Login

E informar o usuário inativado "usuarioinativado@gmail.com"

E informar a senha correta "senhaCorretaDoUsuarioInativo"

E submeter

Então o sistema me apresentará um alerta

E eu não conseguirei logar

@manual

Cenário: Validar login travado por 15 minutos

Dado que acesse o site da EBAC-SHOP

E posteriormente a página de login
Quando eu informar o usuário válido "usuarioativo@gmail.com"
E submeter o formulário com senhas incorretas, por 3 vezes
Então o sistema me apresentará um alerta
E o formulário de login ficará travado por 15 minutos

[US-0003] – API de cupons

```
@automatizado
Cenário: Listar cupons cadastrados
Dado que eu tenha informado um valor válido para "authorization", no header
Quando eu executar o método GET para `baseApiUrl/wc/v3/coupons`
Então a API deverá retornar o status code 200
E listar todos os cupons cadastrados

@automatizado
Cenário: Validar tentativa de listar cupons, passando senha incorreta no header
Dado que eu tenha informado um valor incorreto para "authorization", no header
Quando eu executar o método GET para `baseApiUrl/wc/v3/coupons`
Então a API deverá retornar o status code 500
E na propriedade "code", no corpo da resposta, o valor "incorrect_password"

@manual
Cenário: Listar cupom específico
Dado que eu tenha informado um valor válido para "authorization", no header
Quando eu executar o método GET para `baseApiUrl/wc/v3/coupons/<productId>`
Então a API deverá retornar o status code 200
E listar o produto referente ao id informado no path

@manual
Cenário: Cadastrar cupom
Dado que eu tenha informado um valor válido para "authorization", no header
E informado os seguintes campos obrigatórios:
    | campos          |
    | Código do cupom |
    | Valor           |
    | Tipo do Desconto |
    | Descrição       |
Quando eu executar o método POST para `baseApiUrl/wc/v3/coupons`
Então a API deverá retornar o status code 200
```

4.4 Repositório no Github

- Crie um repositório no github com o nome TCC-EBAC-QE;

- Deixe o repositório publico até a análise dos tutores;
- Neste repositório você deve subir este arquivo e todos os código fontes das automações que criar.
- Referência: Módulo 10
- Link do repositório: <https://github.com/lgc2/TCC-EBAC-QE>

4.5 Testes automatizados

4.5.1 Automação de UI

- Crie um projeto de automação WEB com o framework e a linguagem que preferir
- Justifique a sua escolha através de um comparativo entre ao menos 3 opções de ferramentas e linguagem.
- Crie uma pasta chamada UI para os testes WEB dos casos de teste que forem automatizados
- Utilize ao menos um *Testing Pattern* (à sua escolha) na implementação dos testes.

Entre os frameworks Selenium, Cypress e Playwright, o Cypress foi o escolhido devido sua qualidade e flexibilidade para testes em sistemas Web. A linguagem escolhida foi o Javascript, pois é extremamente utilizada em aplicações Web e amplamente conhecida, além de possuir muitos materiais sobre e comunidade ativa.

Abaixo listo os principais Prós e Contras da utilização do Cypress:

Prós:

- Facilidade de aprendizado;
- Documentação muito rica e comunidade ativa;
- Já é uma tecnologia amplamente utilizada em diversas empresas pelo mundo;
- Facilidade no quesito massa de dados, visto que é muito simples interceptar requisições com o Cypress, e passar o response body desejado na resposta;
- Testes robustos para APIs;
- Possibilita testes de componentes, mesmo que esse não seja o forte da ferramenta;
- Viewport é totalmente configurável, sendo assim podemos testar em diversos breakpoints (Desktop, Tablet, Mobile);
- Tem compatibilidade com o Cucumber, então basta que os cenários sejam escritos em Gherkin que poderemos reutilizá-los nos testes automatizados.

Contras:

- No caso dos testes de API, eles ficarão num projeto diferente do próprio projeto da API;

- Os testes precisam ser feitos sempre no mesmo domínio (exemplo: <https://ebacshop.com.br/...>), mas ainda pode-se validar os links externos fazendo o assert na propriedade href do elemento que se está validando;
- Não é possível testar apps nativos com o Cypress, pois com ele os testes são feitos sempre via Browser.

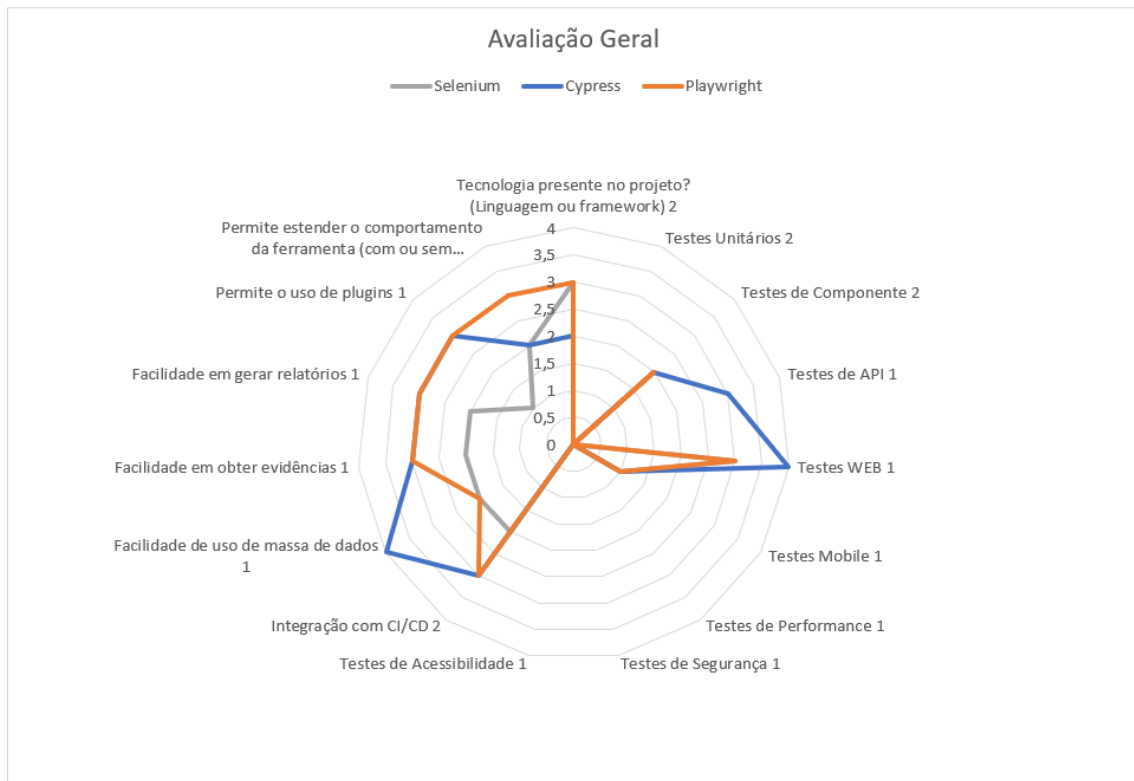


Imagem 1: Avaliação Geral dos frameworks

4.5.2 Automação de API

- Crie uma pasta chamada API para os testes de API dos casos de teste que forem automatizados
- Você deve utilizar a ferramenta Supertest para criar seus testes de API
- Não esqueça de validar os contratos! 😊

4.5.3 Automação Mobile

- Considere para os APPs apenas a funcionalidade de Catálogo de Produtos
- Você pode encontrar os APPs em:
 - *Android*: <https://github.com/EBAC-QE/testes-mobile-ebac-shop/tree/main/app/android>
 - *iOS*: <https://github.com/EBAC-QE/testes-mobile-ebac-shop/tree/ios-tests/app/ios>

- Crie uma pasta chamada Mobile para os testes em aplicativos dos casos de teste que forem automatizados
- Utilize ao menos um *Testing Pattern* (à sua escolha) na implementação dos testes.
- Você deve implementar testes para ao menos uma das plataformas Mobile (*Android* ou *iOS*)
- Observações:
 - Considere todas as boas práticas aprendidas até aqui
 - Não esqueça de implementar a geração de relatórios
- Referência: Módulos 11, 12, 14, 16, 17, 22, 23, 24, 29 e 30






4.6 Integração contínua

- Execute os testes automatizados em integração contínua utilizando o Github Actions
- Referência: Módulo 26

<https://github.com/lgc2/TCC-EBAC-QE> -- branch ci-ui

4.7 Testes de performance

- Usando o K6, implemente um teste de performance em ao menos 2 casos de testes
- Referência: Módulo 28
- 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

5. CONCLUSÃO

Com este trabalho foi possível passar por diversas etapas do desenvolvimento de um software, desde a concepção da história de usuário, passando pelo planejamento dos testes, escrita dos cenários a serem testados, definição de quais deles seriam automatizados, até a execução dos testes automatizados em integração contínua.

Fica muito claro que o Engenheiro de Qualidade de Software tem que ser um profissional multidisciplinar, pois é necessário a ele possuir habilidades técnicas abrangentes, tais como: conhecimentos em diferentes linguagens de programação, frameworks de testes, bibliotecas utilizadas com os frameworks, versionamento de código, Containers, integração e entrega contínua (cultura Devops), testes não funcionais, com Performance, por exemplo, dentre tantas outras.

Mas não é só isso, conforme citado acima, como profissional multidisciplinar o Engenheiro de Qualidade deve possuir também muitas habilidades não técnicas também, as chamadas soft skills. Por participar do produto desde o refinamento do backlog (ou em alguns casos, até antes), esta pessoa precisará possuir uma ótima comunicação, clareza na hora de expor seus pontos de vista. É preciso também possuir inteligência emocional, pois estará em contato direto com diversos outros profissionais com diferentes papéis no desenvolvimento do software.

Cabe ainda ao profissional desta área, espalhar pelo time a cultura da Qualidade como um todo, deixando claro que a responsabilidade pela excelência das entregas é de todos. E para auxiliar nessa missão de conscientização, estão as métricas da qualidade, onde podemos apresentar de uma maneira concreta

a evolução ou não do time, com relação às entregas que estamos fazendo ao cliente.

6. REFERÊNCIAS BIBLIOGRÁFICAS

https://bstqb.org.br/b9/doc/syllabus_ctfl_2018br.pdf

<http://juliodelima.com.br/xdecision/pt>

<https://www.slideshare.net/elias.nogueira/papel-do-qa-na-transformao-gil>

<https://cucumber.io/>

https://www.w3schools.com/cssref/css_selectors.asp

<https://developer.mozilla.org/en-US/docs/Web/HTML/Attributes>

<https://webdriver.io/docs/browserstack-service/>