

**ENGENHEIRO DE QUALIDADE DE SOFTWARE**

Ana Luisa Augusto Barbosa

Análise de Qualidade

São Bernardo do Campo

2024

* **RESUMO**

Este relatório detalha as etapas do processo de testes, contendo desde o planejamento das estratégias desses testes e atribuição de seus cenários, até a aplicação de metodologias ágeis para o gerenciamento do projeto e automoções.

* **SUMÁRIO**

***1.*** ***RESUMO 2***

***2.*** ***SUMÁRIO 3***

***3.*** ***INTRODUÇÃO 4***

***4.*** ***O PROJETO 5***

**4.1** **Estratégia de teste 6**

**4.2** **Critérios de aceitação 6**

**4.3** **Casos de testes 6**

**4.4** **Repositório no Github 7**

**4.5** **Testes automatizados 7**

**4.6** **Integração contínua 8**

**4.7** **Testes de performance 8**

***5.*** ***CONCLUSÃO 9***

***6.*** ***REFERÊNCIAS BIBLIOGRÁFICAS 9***

* **INTRODUÇÃO**

Tendo com objetivo garantir a eficiência e proporcionar uma excelente experiência na Ebac Shop, mantendo a plataforma estável e de alta qualidade. Realizei um planejamento estratégico, mantendo o foco em um time ágil e comprometido. Selecionei ferramentas sólidas do mercado para minimizar o impacto no ambiente de produção, reduzir bugs e assegurar a performance do sistema. O uso de recursos adequados é essencial para criar uma plataforma confiável e robusta, oferecendo uma experiência de navegação satisfatória e segura para os usuários.

* **O PROJETO**
* **Estratégia de teste**



* **Critérios de aceitação**

[US-0001] – Adicionar item ao carrinho

* [US-0002] – Login na plataforma
* [US-0003] – API de cupons
* 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
* Painel Minha Conta
* Meus Pedidos
* Endereços
* Detalhes da Conta
* Referência: Módulo 8
* **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
* **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: <cole o link aqui>
* **Testes automatizados**
* 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.
* 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!
* 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
* **Integração contínua**
* Execute os testes automatizados em integração contínua utilizando o Github Actions
* Referência: Módulo 26
* **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



* **CONCLUSÃO**

Coloque sua experiência na realização do trabalho, o que aprendeu, quais lições pode aplicar em sua vida profissional etc.

* **REFERÊNCIAS BIBLIOGRÁFICAS**

Seguir regras ABNT